To all our customers

## Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

## Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.

# H8S/2655 Series

# H8S/2655

## HD6432655, HD6472655,

# H8S/2653

## HD6432653

Hardware Manual

ADE-602-094C
Rev. 4.0
3/6/03
Hitachi, Ltd.
MC-Setsu

# Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

# Preface

The H8S/2655 Series is a series of high-performance microcontrollers with a 32-bit H8S/2600 CPU core, and a set of on-chip supporting functions required for system configuration.

The H8S/2600 CPU can execute basic instructions in one state, and is provided with sixteen 16-bit general registers with a 32-bit internal configuration, and a concise and optimized instruction set. The CPU can handle a 16 Mbyte linear address space (architecturally 4 Gbytes). Programs based on the high-level language C can also be run efficiently.

The address space is divided into eight areas. The data bus width and access states can be selected for each of these areas, and various kinds of memory can be connected fast and easily.

On-chip memory consists of large-capacity ROM and RAM. PROM (ZTAT™*) and mask ROM versions are available, providing a quick and flexible response to conditions from ramp-up through full-scale volume production, even for applications with frequently changing specifications.

On-chip supporting functions include a 16-bit timer pulse unit (TPU), programmable pulse generator (PPG), 8-bit timers, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

In addition, an on-chip DMA controller (DMAC) and data transfer controller (DTC) are provided, enabling high-speed data transfer without CPU intervention.

Use of the H8S/2655 Series enables compact, high-performance systems to be implemented easily.

This manual describes the hardware of the H8S/2655 Series. Refer to the H8S/2600 Series and H8S/2000 Series Programming Manual for a detailed description of the instruction set.

Note: * ZTAT is a trademark of Hitachi, Ltd.

# Main Revisions and Additions in This Edition

| Page | Section | Revision Contents |
|---|---|---|
| 144 | 6.2.6. Memory Control Register (MCR): description of Bit 5 | Description amendments |
| 169 | Figure 6-14 Example of Wait State Insertion Timing | Figure amended |
| 211 | Figure 6-47 Bus-Released State Transition Timing | Figure amended |
| 265 | Table 7-9 Register Functions in Single Address Mode | Table amendments |
| 277 | 7.5.8 DMAC Activation Sources<br>Single Address Mode | Description amendments |
| 430 | 10.2.1 Timer Control Register (TCR) | Description amendments |
| 435 | 10.2.2 Timer Mode Register (TMDR) | Descriptive text added |
| 437 | 10.2.3 Timer I/O Control Register (TIOR) | Note amended |
| 459 | 10.2.8 Timer Start Register (TSTR) | Description amendments |
| 511 | 10.7 Usage Notes: Contention between TCNT Write and Overflow/Underflow | Descriptive text added |
| 511 | 10.7 Usage Notes: Interrupts and Module Stop Mode | addition |
| 605 | 14.2.9 Smart Card Mode Register (SCMR): description of Bit 3 | Descriptive text added |
| 629 | Figure 14-15 Sample SCI Initialization Flowchart | Note added |
| 635 | Figure 14-20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations | Note amended |
| 649 | 15.2.3 Serial Mode Register (SMR): description of Bit 7 | Amended |
| 651 | 15.2.4 Serial Control Register (SCR) | Amended |
| 715 | 19.2.1 Bus Control Register L (BCRL): description of Bits 4 and 3 | Added |
| 718 | Figure 19-2 Wiring of Socket Adapter | Figure amended |
| 802 to 804 | Table A.1 Instruction Set | Table amendments |
| 1005 | Figure C-1 (a) Port1 Block Diagram (Pins $P1_0$ and $P1_1$) | Note deleted |
| 1006 | Figure C-1 (b) Port1 Block Diagram (Pins $P1_2$, $P1_3$, $P1_5$, and $P1_7$) | Note deleted |
| 1007 | Figure C-1 (c) Port1 Block Diagram (Pins $P1_4$ and $P1_6$) | Note deleted |
| 1008 | Figure C-2 (a) Port2 Block Diagram (Pins $P2_0$ and $P2_1$) | Note deleted |
| 1009 | Figure C-2 (b) Port2 Block Diagram (Pins $P2_2$ and $P2_4$) | Note deleted |
| 1010 | Figure C-2 (c) Port2 Block Diagram (Pins $P2_3$ and $P2_5$) | Note deleted |
| 1011 | Figure C-2 (d) Port2 Block Diagram (Pins $P2_6$ and $P2_7$) | Note deleted |
| 1026 | Figure C-7 (a) PortA Block Diagram (Pins $PA_0$ to $PA_3$) | Figure amended |

| Page | Section | Revision Contents |
|---|---|---|
| 1027 | Figure C-7 (b) PortA Block Diagram (Pin $PA_4$) | Figure amended |
| 1028 | Figure C-7 (c) PortA Block Diagram (Pins $PA_5$ to $PA_7$) | Figure amended |
| 1031 | Figure C-10 PortD Block Diagram (Pin $PD_n$) | Figure amended |
| 1032 | Figure C-11 PortE Block Diagram (Pin $PE_n$) | Figure amended |
| 1044,<br>1047 | Table D-1 I/O Port States in Each Processing State<br>Port A<br>Port $PF_1$ | Table added |

# Contents


Section 1 Overview ..... 1
1.1 Overview ..... 1
1.2 Block Diagram ..... 5
1.3 Pin Description ..... 6
1.3.1 Pin Arrangement ..... 6
1.3.2 Pin Functions in Each Operating Mode ..... 8
1.3.3 Pin Functions ..... 13

Section 2 CPU ..... 21
2.1 Overview ..... 21
2.1.1 Features ..... 21
2.1.2 Differences from H8/300 CPU ..... 22
2.1.3 Differences from H8/300H CPU ..... 23
2.2 CPU Operating Modes ..... 24
2.3 Address Space ..... 29
2.4 Register Configuration ..... 30
2.4.1 Overview ..... 30
2.4.2 General Registers ..... 31
2.4.3 Control Registers ..... 32
2.4.4 Initial Register Values ..... 34
2.5 Data Formats ..... 35
2.5.1 General Register Data Formats ..... 35
2.5.2 Memory Data Formats ..... 37
2.6 Instruction Set ..... 38
2.6.1 Overview ..... 38
2.6.2 Instructions and Addressing Modes ..... 39
2.6.3 Table of Instructions Classified by Function ..... 41
2.6.4 Basic Instruction Formats ..... 50
2.7 Addressing Modes and Effective Address Calculation ..... 52
2.7.1 Addressing Mode ..... 52
2.7.2 Effective Address Calculation ..... 55
2.8 Processing States ..... 59
2.8.1 Overview ..... 59
2.8.2 Reset State ..... 60
2.8.3 Exception-Handling State ..... 61
2.8.4 Program Execution State ..... 64
2.8.5 Bus-Released State ..... 64
2.8.6 Power-Down State ..... 64
2.9 Basic Timing ..... 65

2.9.1 Overview ..... 65
2.9.2 On-Chip Memory (ROM, RAM) ..... 65
2.9.3 On-Chip Supporting Module Access Timing ..... 67
2.9.4 External Address Space Access Timing ..... 68

Section 3 MCU Operating Modes ..... 69
3.1 Overview ..... 69
3.1.1 Operating Mode Selection ..... 69
3.1.2 Register Configuration ..... 70
3.2 Register Descriptions ..... 71
3.2.1 Mode Control Register (MDCR) ..... 71
3.2.2 System Control Register (SYSCR) ..... 71
3.3 Operating Mode Descriptions ..... 73
3.3.1 Mode 1 ..... 73
3.3.2 Mode 2 ..... 73
3.3.3 Mode 3 ..... 73
3.3.4 Mode 4 ..... 73
3.3.5 Mode 5 ..... 74
3.3.6 Mode 6 ..... 74
3.3.7 Mode 7 ..... 74
3.4 Pin Functions in Each Operating Mode ..... 75
3.5 Memory Map in Each Operating Mode ..... 76

Section 4 Exception Handling ..... 79
4.1 Overview ..... 79
4.1.1 Exception Handling Types and Priority ..... 79
4.1.2 Exception Handling Operation ..... 79
4.1.3 Exception Vector Table ..... 80
4.2 Reset ..... 82
4.2.1 Overview ..... 82
4.2.2 Reset Types ..... 82
4.2.3 Reset Sequence ..... 83
4.2.4 Interrupts after Reset ..... 84
4.3 Traces ..... 85
4.4 Interrupts ..... 86
4.5 Trap Instruction ..... 87
4.6 Stack Status after Exception Handling ..... 88
4.7 Notes on Use of the Stack ..... 89

Section 5 Interrupt Controller ..... 91
5.1 Overview ..... 91
5.1.1 Features ..... 91
5.1.2 Block Diagram ..... 92

5.1.3 Pin Configuration ..... 92
5.1.4 Register Configuration ..... 93
5.2 Register Descriptions ..... 94
5.2.1 System Control Register (SYSCR) ..... 94
5.2.2 Interrupt Control Registers A to C (ICRA to ICRC) ..... 95
5.2.3 Interrupt Priority Registers A to K (IPRA to IPRK) ..... 95
5.2.4 IRQ Enable Register (IER) ..... 97
5.2.5 IRQ Sense Control Registers H and L (ISCRH, ISCRL) ..... 97
5.2.6 IRQ Status Register (ISR) ..... 98
5.3 Interrupt Sources ..... 100
5.3.1 External Interrupts ..... 100
5.3.2 Internal Interrupts ..... 102
5.3.3 Interrupt Exception Handling Vector Table ..... 102
5.4 Interrupt Operation ..... 106
5.4.1 Interrupt Control Modes and Interrupt Operation ..... 106
5.4.2 Interrupt Control Mode 0 ..... 110
5.4.3 Interrupt Control Mode 1 ..... 112
5.4.4 Interrupt Control Mode 2 ..... 115
5.4.5 Interrupt Control Mode 3 ..... 117
5.4.6 Interrupt Exception Handling Sequence ..... 119
5.4.7 Interrupt Response Times ..... 121
5.5 Usage Notes ..... 122
5.5.1 Contention between Interrupt Generation and Disabling ..... 122
5.5.2 Instructions that Disable Interrupts ..... 123
5.5.3 Times when Interrupts are Disabled ..... 123
5.5.4 Interrupts during Execution of EEPMOV Instruction ..... 123
5.6 DTC and DMAC Activation by Interrupt ..... 124
5.6.1 Overview ..... 124
5.6.2 Block Diagram ..... 124
5.6.3 Operation ..... 125

Section 6 Bus Controller ..... 127
6.1 Overview ..... 127
6.1.1 Features ..... 127
6.1.2 Block Diagram ..... 129
6.1.3 Pin Configuration ..... 130
6.1.4 Register Configuration ..... 132
6.2 Register Descriptions ..... 133
6.2.1 Bus Width Control Register (ABWCR) ..... 133
6.2.2 Access State Control Register (ASTCR) ..... 134
6.2.3 Wait Control Registers H and L (WCRH, WCRL) ..... 135
6.2.4 Bus Control Register H (BCRH) ..... 139
6.2.5 Bus Control Register L (BCRL) ..... 141

6.2.6 Memory Control Register (MCR) ..... 144
6.2.7 DRAM Control Register (DRAMCR) ..... 147
6.2.8 Refresh Timer/Counter (RTCNT) ..... 150
6.2.9 Refresh Time Constant Register (RTCOR) ..... 150
6.3 Overview of Bus Control ..... 151
6.3.1 Area Partitioning ..... 151
6.3.2 Bus Specifications ..... 152
6.3.3 Memory Interfaces ..... 153
6.3.4 Advanced Mode ..... 154
6.3.5 Areas in Normal Mode ..... 155
6.3.6 Chip Select Signals ..... 156
6.4 Basic Bus Interface ..... 157
6.4.1 Overview ..... 157
6.4.2 Data Size and Data Alignment ..... 157
6.4.3 Valid Strobes ..... 159
6.4.4 Basic Timing ..... 160
6.4.5 Wait Control ..... 168
6.5 DRAM Interface ..... 170
6.5.1 Overview ..... 170
6.5.2 Setting DRAM Space ..... 170
6.5.3 Address Multiplexing ..... 170
6.5.4 Data Bus ..... 171
6.5.5 Pins Used for DRAM Interface ..... 172
6.5.6 Basic Timing ..... 173
6.5.7 Precharge State Control ..... 174
6.5.8 Wait Control ..... 175
6.5.9 Byte Access Control ..... 177
6.5.10 Burst Operation ..... 181
6.5.11 Caution Concerning 2-CAS System ..... 184
6.5.12 Refresh Control ..... 185
6.6 Pseudo-SRAM Interface ..... 189
6.6.1 Overview ..... 189
6.6.2 Setting PSRAM Space ..... 189
6.6.3 Data Bus ..... 189
6.6.4 Pins Used for PSRAM Interface ..... 190
6.6.5 Basic Timing ..... 191
6.6.6 Precharge State Control ..... 192
6.6.7 Wait Control ..... 193
6.6.8 Burst Operation ..... 195
6.6.9 Refresh Control ..... 198
6.6.10 Power-On Sequence ..... 199
6.7 DMAC Single Address Mode and DRAM/PSRAM Interface ..... 200
6.7.1 When DDS=1 ..... 200

6.7.2 When DDS=0 ... 201
6.8 Burst ROM Interface ... 202
6.8.1 Overview ... 202
6.8.2 Basic Timing ... 202
6.8.3 Wait Control ... 204
6.9 Idle Cycle ... 205
6.9.1 Operation ... 205
6.9.2 Pin States in Idle Cycle ... 207
6.10 Write Data Buffer Function ... 208
6.11 Bus Release ... 209
6.11.1 Overview ... 209
6.11.2 Operation ... 209
6.11.3 Pin States in External Bus Released State ... 210
6.11.4 Transition Timing ... 211
6.12 Bus Arbitration ... 212
6.12.1 Overview ... 212
6.12.2 Operation ... 212
6.12.3 Bus Transfer Timing ... 213
6.12.4 External Bus Release Usage Note ... 213
6.13 Resets and the Bus Controller ... 214

Section 7 DMA Controller ... 215
7.1 Overview ... 215
7.1.1 Features ... 215
7.1.2 Block Diagram ... 216
7.1.3 Overview of Functions ... 217
7.1.4 Pin Configuration ... 219
7.1.5 Register Configuration ... 220
7.2 Register Descriptions (1) (Short Address Mode) ... 221
7.2.1 Memory Address Registers (MAR) ... 222
7.2.2 I/O Address Register (IOAR) ... 223
7.2.3 Execute Transfer Count Register (ETCR) ... 223
7.2.4 DMA Control Register (DMACR) ... 224
7.2.5 DMA Band Control Register (DMABCR) ... 229
7.3 Register Descriptions (2) (Full Address Mode) ... 235
7.3.1 Memory Address Register (MAR) ... 235
7.3.2 I/O Address Register (IOAR) ... 235
7.3.3 Execute Transfer Count Register (ETCR) ... 236
7.3.4 DMA Control Register (DMACR) ... 237
7.3.5 DMA Band Control Register (DMABCR) ... 241
7.4 Register Descriptions (3) ... 247
7.4.1 DMA Write Enable Register (DMAWER) ... 247
7.4.2 DMA Terminal Control Register (DMATCR) ... 250

7.4.3 Module Stop Control Register (MSTPCR) ... 251
7.5 Operation ... 252
7.5.1 Transfer Modes ... 252
7.5.2 Sequential Mode ... 255
7.5.3 Idle Mode ... 258
7.5.4 Repeat Mode ... 261
7.5.5 Single Address Mode ... 265
7.5.6 Normal Mode ... 268
7.5.7 Block Transfer Mode ... 271
7.5.8 DMAC Activation Sources ... 277
7.5.9 Basic DMAC Bus Cycles ... 280
7.5.10 DMAC Bus Cycles (Dual Address Mode) ... 281
7.5.11 DMAC Bus Cycles (Single Address Mode) ... 289
7.5.12 Write Data Buffer Function ... 295
7.5.13 DMAC Multi-Channel Operation ... 296
7.5.14 Relation Between External Bus Requests, Refresh Cycles, the DTC, and the DMAC ... 298
7.5.15 NMI Interrupts and DMAC ... 299
7.5.16 Forced Termination of DMAC Operation ... 300
7.5.17 Clearing Full Address Mode ... 301
7.6 Interrupts ... 302
7.7 Usage Notes ... 303

Section 8 Data Transfer Controller ... 307
8.1 Overview ... 307
8.1.1 Features ... 307
8.1.2 Block Diagram ... 308
8.1.3 Register Configuration ... 309
8.2 Register Descriptions ... 310
8.2.1 DTC Mode Register A (MRA) ... 310
8.2.2 DTC Mode Register B (MRB) ... 312
8.2.3 DTC Source Address Register (SAR) ... 313
8.2.4 DTC Destination Address Register (DAR) ... 313
8.2.5 DTC Transfer Count Register A (CRA) ... 313
8.2.6 DTC Transfer Count Register B (CRB) ... 314
8.2.7 DTC Enable Registers (DTCER) ... 314
8.2.8 DTC Vector Register (DTVECR) ... 315
8.2.9 Module Stop Control Register (MSTPCR) ... 316
8.3 Operation ... 317
8.3.1 Overview ... 317
8.3.2 Activation Sources ... 319
8.3.3 DTC Vector Table ... 320
8.3.4 Location of Register Information in Address Space ... 323

8.3.5 Normal Mode ..... 324
8.3.6 Repeat Mode ..... 325
8.3.7 Block Transfer Mode ..... 326
8.3.8 Chain Transfer ..... 328
8.3.9 Operation Timing ..... 329
8.3.10 Number of DTC Execution States ..... 330
8.3.11 Procedures for Using DTC ..... 332
8.3.12 Examples of Use of the DTC ..... 333
8.4 Interrupts ..... 336
8.5 Usage Notes ..... 336

Section 9 I/O Ports ..... 337
9.1 Overview ..... 337
9.2 Port 1 ..... 343
9.2.1 Overview ..... 343
9.2.2 Register Configuration ..... 344
9.2.3 Pin Functions ..... 346
9.3 Port 2 ..... 354
9.3.1 Overview ..... 354
9.3.2 Register Configuration ..... 355
9.3.3 Pin Functions ..... 357
9.4 Port 3 ..... 365
9.4.1 Overview ..... 365
9.4.2 Register Configuration ..... 365
9.4.3 Pin Functions ..... 368
9.5 Port 4 ..... 370
9.5.1 Overview ..... 370
9.5.2 Register Configuration ..... 371
9.5.3 Pin Functions ..... 371
9.6 Port 5 ..... 372
9.6.1 Overview ..... 372
9.6.2 Register Configuration ..... 373
9.6.3 Pin Functions ..... 375
9.7 Port 6 ..... 376
9.7.1 Overview ..... 376
9.7.2 Register Configuration ..... 377
9.7.3 Pin Functions ..... 379
9.8 Port A ..... 381
9.8.1 Overview ..... 381
9.8.2 Register Configuration ..... 382
9.8.3 Pin Functions ..... 385
9.8.4 MOS Input Pull-Up Function ..... 388
9.9 Port B ..... 389

9.9.1 Overview ... 389
9.9.2 Register Configuration ... 390
9.9.3 Pin Functions ... 392
9.9.4 MOS Input Pull-Up Function ... 394
9.10 Port C ... 395
9.10.1 Overview ... 395
9.10.2 Register Configuration ... 396
9.10.3 Pin Functions ... 398
9.10.4 MOS Input Pull-Up Function ... 400
9.11 Port D ... 401
9.11.1 Overview ... 401
9.11.2 Register Configuration ... 402
9.11.3 Pin Functions ... 404
9.11.4 MOS Input Pull-Up Function ... 405
9.12 Port E ... 406
9.12.1 Overview ... 406
9.12.2 Register Configuration ... 407
9.12.3 Pin Functions ... 409
9.12.4 MOS Input Pull-Up Function ... 410
9.13 Port F ... 411
9.13.1 Overview ... 411
9.13.2 Register Configuration ... 412
9.13.3 Pin Functions ... 414
9.14 Port G ... 416
9.14.1 Overview ... 416
9.14.2 Register Configuration ... 417
9.14.3 Pin Functions ... 419

Section 10 16-Bit Timer Pulse Unit (TPU) ... 421
10.1 Overview ... 421
10.1.1 Features ... 421
10.1.2 Block Diagram ... 425
10.1.3 Pin Configuration ... 426
10.1.4 Register Configuration ... 428
10.2 Register Descriptions ... 430
10.2.1 Timer Control Register (TCR) ... 430
10.2.2 Timer Mode Register (TMDR) ... 435
10.2.3 Timer I/O Control Register (TIOR) ... 437
10.2.4 Timer Interrupt Enable Register (TIER) ... 450
10.2.5 Timer Status Register (TSR) ... 453
10.2.6 Timer Counter (TCNT) ... 457
10.2.7 Timer General Register (TGR) ... 458
10.2.8 Timer Start Register (TSTR) ... 459

10.2.9 Timer Synchro Register (TSYR) ..... 460
10.2.10 Module Stop Control Register (MSTPCR) ..... 461
10.3 Interface to Bus Master ..... 462
10.3.1 16-Bit Registers ..... 462
10.3.2 8-Bit Registers ..... 462
10.4 Operation ..... 464
10.4.1 Overview ..... 464
10.4.2 Basic Functions ..... 465
10.4.3 Synchronous Operation ..... 471
10.4.4 Buffer Operation ..... 473
10.4.5 Cascaded Operation ..... 477
10.4.6 PWM Modes ..... 479
10.4.7 Phase Counting Mode ..... 484
10.5 Interrupts ..... 491
10.5.1 Interrupt Sources and Priorities ..... 491
10.5.2 DTC/DMAC Activation ..... 493
10.5.3 A/D Converter Activation ..... 493
10.6 Operation Timing ..... 494
10.6.1 Input/Output Timing ..... 494
10.6.2 Interrupt Signal Timing ..... 498
10.7 Usage Notes ..... 502

Section 11 Programmable Pulse Generator (PPG) ..... 513
11.1 Overview ..... 513
11.1.1 Features ..... 513
11.1.2 Block Diagram ..... 514
11.1.3 Pin Configuration ..... 515
11.1.4 Registers ..... 516
11.2 Register Descriptions ..... 517
11.2.1 Next Data Enable Registers H and L (NDERH, NDERL) ..... 517
11.2.2 Output Data Registers H and L (PODRH, PODRL) ..... 518
11.2.3 Next Data Registers H and L (NDRH, NDRL) ..... 519
11.2.4 Notes on NDR Access ..... 520
11.2.5 PPG Output Control Register (PCR) ..... 522
11.2.6 PPG Output Mode Register (PMR) ..... 524
11.2.7 Port 1 Data Direction Register (P1DDR) ..... 527
11.2.8 Port 2 Data Direction Register (P2DDR) ..... 527
11.2.9 Module Stop Control Register (MSTPCR) ..... 528
11.3 Operation ..... 529
11.3.1 Overview ..... 529
11.3.2 Output Timing ..... 530
11.3.3 Normal Pulse Output ..... 531
11.3.4 Non-Overlapping Pulse Output ..... 533

11.3.5 Inverted Pulse Output ..... 536
11.3.6 Pulse Output Triggered by Input Capture ..... 537
11. 4 Usage Notes ..... 538
11.4.1 Operation of Pulse Output Pins ..... 538
11.4.2 Note on Non-Overlapping Output ..... 538

Section 12 8-Bit Timers ..... 541
12.1 Overview ..... 541
12.1.1 Features ..... 541
12.1.2 Block Diagram ..... 542
12.1.3 Pin Configuration ..... 543
12.1.4 Register Configuration ..... 543
12.2 Register Descriptions ..... 544
12.2.1 Timer Counters 0 and 1 (TCNT0, TCNT1) ..... 544
12.2.2 Time Constant Registers A0 and A1 (TCORA0, TCORA1) ..... 544
12.2.3 Time Constant Registers B0 and B1 (TCORB0, TCORB1) ..... 545
12.2.4 Time Control Registers 0 and 1 (TCR0, TCR1) ..... 545
12.2.5 Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1) ..... 548
12.2.6 Module Stop Control Register (MSTPCR) ..... 551
12.3 Operation ..... 552
12.3.1 TCNT Incrementation Timing ..... 552
12.3.2 Compare Match Timing ..... 553
12.3.3 Timing of External RESET on TCNT ..... 555
12.3.4 Timing of Overflow Flag (OVF) Setting ..... 555
12.3.5 Operation with Cascaded Connection ..... 556
12.4 Interrupt Sources ..... 557
12.5 Sample Application ..... 558
12.6 Usage Notes ..... 559
12.6.1 Contention between TCNT Write and Clear ..... 559
12.6.2 Contention between TCNT Write and Increment ..... 560
12.6.3 Contention between TCOR Write and Compare Match ..... 561
12.6.4 Contention between Compare Matches A and B ..... 562
12.6.5 Switching of Internal Clocks and TCNT Operation ..... 562

Section 13 Watchdog Timer ..... 565
13.1 Overview ..... 565
13.1.1 Features ..... 565
13.1.2 Block Diagram ..... 566
13.1.3 Pin Configuration ..... 567
13.1.4 Register Configuration ..... 567
13.2 Register Descriptions ..... 568
13.2.1 Timer Counter (TCNT) ..... 568
13.2.2 Timer Control/Status Register (TCSR) ..... 568

13.2.3 Reset Control/Status Register (RSTCSR) ..... 570
13.2.4 Notes on Register Access ..... 572
13.3 Operation ..... 574
13.3.1 Watchdog Timer Operation ..... 574
13.3.2 Interval Timer Operation ..... 575
13.3.3 Timing of Setting Overflow Flag (OVF) ..... 575
13.3.4 Timing of Setting of Watchdog Timer Overflow Flag (WOVF) ..... 576
13.4 Interrupts ..... 577
13.5 Usage Notes ..... 577
13.5.1 Contention between Timer Counter (TCNT) Write and Increment ..... 577
13.5.2 Changing Value of CKS2 to CKS0 ..... 577
13.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode ..... 578
13.5.4 System Reset by $\overline{\text{WDTOVF}}$ Signal ..... 578
13.5.5 Internal Reset in Watchdog Timer Mode ..... 578

Section 14 Serial Communication Interface (SCI) ..... 579
14.1 Overview ..... 579
14.1.1 Features ..... 579
14.1.2 Block Diagram ..... 581
14.1.3 Pin Configuration ..... 582
14.1.4 Register Configuration ..... 583
14.2 Register Descriptions ..... 584
14.2.1 Receive Shift Register (RSR) ..... 584
14.2.2 Receive Data Register (RDR) ..... 584
14.2.3 Transmit Shift Register (TSR) ..... 585
14.2.4 Transmit Data Register (TDR) ..... 585
14.2.5 Serial Mode Register (SMR) ..... 586
14.2.6 Serial Control Register (SCR) ..... 589
14.2.7 Serial Status Register (SSR) ..... 593
14.2.8 Bit Rate Register (BRR) ..... 596
14.2.9 Smart Card Mode Register (SCMR) ..... 605
14.2.10 Module Stop Control Register (MSTPCR) ..... 606
14.3 Operation ..... 607
14.3.1 Overview ..... 607
14.3.2 Operation in Asynchronous Mode ..... 609
14.3.3 Multiprocessor Communication Function ..... 620
14.3.4 Operation in Clocked Synchronous Mode ..... 627
14.4 SCI Interrupts ..... 636
14.5 Usage Notes ..... 638

Section 15 Smart Card Interface ..... 643
15.1 Overview ..... 643
15.1.1 Features ..... 643

15.1.2 Block Diagram ..... 644
15.1.3 Pin Configuration ..... 645
15.1.4 Register Configuration ..... 646
15.2 Register Descriptions ..... 647
15.2.1 Smart Card Mode Register (SCMR) ..... 647
15.2.2 Serial Status Register (SSR) ..... 648
15.2.3 Serial Mode Register (SMR) ..... 649
15.2.4 Serial Control Register (SCR) ..... 651
15.3 Operation ..... 652
15.3.1 Overview ..... 652
15.3.2 Pin Connections ..... 652
15.3.3 Data Format ..... 654
15.3.4 Register Settings ..... 656
15.3.5 Clock ..... 658
15.3.6 Data Transfer Operations ..... 660
15.3.7 Example of Use of Software Standby Mode ..... 666
15.3.8 Powering On ..... 667
15.4 Usage Notes ..... 668

Section 16 A/D Converter ..... 671
16.1 Overview ..... 671
16.1.1 Features ..... 671
16.1.2 Block Diagram ..... 672
16.1.3 Pin Configuration ..... 672
16.1.4 Register Configuration ..... 673
16.2 Register Descriptions ..... 674
16.2.1 A/D Data Registers A to H (ADDRA to ADDRH) ..... 674
16.2.2 A/D Control/Status Register (ADCSR) ..... 675
16.2.3 A/D Control Register (ADCR) ..... 677
16.2.4 Module Stop Control Register (MSTPCR) ..... 680
16.3 Interface to Bus Master ..... 681
16.4 Operation ..... 682
16.4.1 Select Single Mode ..... 683
16.4.2 Select Scan Mode ..... 684
16.4.3 Group Single Mode ..... 685
16.4.4 Group Scan Mode ..... 686
16.4.5 Buffer Operation ..... 687
16.4.6 Simultaneous Sampling Operation ..... 690
16.4.7 Conversion Start Modes ..... 691
16.4.8 Starting Conversion by External Input ..... 694
16.4.9 A/D Conversion Time ..... 695
16.5 Interrupts ..... 698
16.6 Usage Notes ..... 699

Section 17 D/A Converter ..... 701
17.1 Overview ..... 701
17.1.1 Features ..... 701
17.1.2 Block Diagram ..... 702
17.1.3 Pin Configuration ..... 703
17.1.4 Register Configuration ..... 703
17.2 Register Descriptions ..... 704
17.2.1 D/A Data Registers 0 and 1 (DADR0, DADR1) ..... 704
17.2.2 D/A Control Register (DACR) ..... 704
17.2.3 Module Stop Control Register (MSTPCR) ..... 706
17.3 Operation ..... 707

Section 18 RAM ..... 709
18.1 Overview ..... 709
18.1.1 Block Diagram ..... 709
18.1.2 Register Configuration ..... 710
18.2 Register Descriptions ..... 711
18.2.1 System Control Register (SYSCR) ..... 711
18.3 Operation ..... 712
18.4 Usage Notes ..... 712

Section 19 ROM ..... 713
19.1 Overview ..... 713
19.1.1 Block Diagram ..... 714
19.1.2 Register Configuration ..... 714
19.2 Register Descriptions ..... 715
19.2.1 Bus Control Register L (BCRL) ..... 715
19.3 Operation ..... 716
19.4 PROM Mode ..... 717
19.4.1 PROM Mode Setting ..... 717
19.4.2 Socket Adapter and Memory Map ..... 717
19.5 Programming ..... 720
19.5.1 Overview ..... 720
19.5.2 Programming and Verification ..... 720
19.5.3 Programming Precautions ..... 725
19.5.4 Reliability of Programmed Data ..... 726

Section 20 Clock Pulse Generator ..... 727
20.1 Overview ..... 727
20.1.1 Block Diagram ..... 727
20.1.2 Register Configuration ..... 728
20.2 Register Descriptions ..... 729

20.2.1 System Clock Control Register (SCKCR) ... 729
20.3 Oscillator ... 730
20.3.1 Connecting a Crystal Resonator ... 730
20.3.2 External Clock Input ... 732
20.4 Duty Adjustment Circuit ... 734
20.5 Medium-Speed Clock Divider ... 734
20.6 Bus Master Clock Selection Circuit ... 734

Section 21 Power-Down Modes ... 735
21.1 Overview ... 735
21.1.1 Register Configuration ... 736
21.2 Register Descriptions ... 737
21.2.1 Standby Control Register (SBYCR) ... 737
21.2.2 System Clock Control Register (SCKCR) ... 738
21.2.3 Module Stop Control Register (MSTPCR) ... 739
21.3 Medium-Speed Mode ... 740
21.4 Sleep Mode ... 742
21.5 Module Stop Mode ... 743
21.5.1 Module Stop Mode ... 743
21.5.2 Usage Notes ... 744
21.6 Software Standby Mode ... 745
21.6.1 Software Standby Mode ... 745
21.6.2 Clearing Software Standby Mode ... 745
21.6.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode ... 746
21.6.4 Software Standby Mode Application Example ... 746
21.6.5 Usage Notes ... 747
21.7 Hardware Standby Mode ... 748
21.7.1 Hardware Standby Mode ... 748
21.7.2 Hardware Standby Mode Timing ... 749
21.8 ø Clock Output Disabling Function ... 750

Section 22 Electrical Characteristics ... 751
22.1 Absolute Maximum Ratings ... 751
22.2 DC Characteristics ... 752
22.3 AC Characteristics ... 758
22.3.1 Clock Timing ... 759
22.3.2 Control Signal Timing ... 761
22.3.3 Bus Timing ... 763
22.3.4 DMAC Timing ... 775
22.3.5 Timing of On-Chip Supporting Modules ... 779
22.4 A/D Conversion Characteristics ... 784
22.5 D/A Conversion Characteristics ... 785

22.6 Usage Notes ..... 785

Appendix A Instruction Set ..... 787
A.1 Instruction List ..... 787
A.2 Instruction Codes ..... 812
A.3 Operation Code Map ..... 827
A.4 Number of States Required for Instruction Execution ..... 831
A.5 Bus States During Instruction Execution ..... 842
A.6 Condition Code Modification ..... 856

Appendix B Internal I/O Register ..... 862
B.1 Addresses ..... 862
B.2 Functions ..... 873

Appendix C I/O Port Block Diagrams ..... 1005
C.1 Port 1 Block Diagram ..... 1005
C.2 Port 2 Block Diagram ..... 1008
C.3 Port 3 Block Diagram ..... 1012
C.4 Port 4 Block Diagram ..... 1015
C.5 Port 5 Block Diagram ..... 1016
C.6 Port 6 Block Diagram ..... 1020
C.7 Port A Block Diagram ..... 1026
C.8 Port B Block Diagram ..... 1029
C.9 Port C Block Diagram ..... 1030
C.10 Port D Block Diagram ..... 1031
C.11 Port E Block Diagram ..... 1032
C.12 Port F Block Diagram ..... 1033
C.13 Port G Block Diagram ..... 1041

Appendix D Pin States ..... 1044
D.1 Port States in Each Mode ..... 1044

Appendix E Timing of Transition to and Recovery from Hardware Standby Mode ..... 1049

Appendix F Product Code Lineup ..... 1050

Appendix G Package Dimensions ..... 1051

# Section 1 Overview

## 1.1 Overview

The H8S/2655 Series is a series of microcomputers (MCUs: microcomputer units), built around the H8S/2600 CPU, employing Hitachi's proprietary architecture, and equipped with peripheral functions on-chip.

The H8S/2600 CPU has an internal 32-bit architecture, is provided with sixteen 16-bit general registers and a concise, optimized instruction set designed for high-speed operation, and can address a 16-Mbyte linear address space. The instruction set is upward-compatible with H8/300 and H8/300H CPU instructions at the object-code level, facilitating migration from the H8/300, H8/300L, or H8/300H Series.

On-chip peripheral functions required for system configuration include DMA controller (DMAC) and data transfer controller (DTC) bus masters, ROM and RAM memory, a16-bit timer-pulse unit (TPU), programmable pulse generator (PPG), 8-bit timer, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

The on-chip ROM is either PROM (ZTAT™*) or mask ROM, with a capacity of 128 or 64 kbytes. ROM is connected to the CPU via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching has been speeded up, and processing speed increased.

Seven operating modes, modes 1 to 7, are provided, and there is a choice of address space and single-chip mode or external expansion mode.

The features of the H8S/2655 Series are shown in Table 1-1.

Note: * ZTAT™ is a trademark of Hitachi, Ltd.

**Table 1-1 Overview**

| Item | Specification |
|---|---|
| CPU | • General-register machine<br>— Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)<br>• High-speed operation suitable for realtime control<br>— Maximum clock rate: 20 MHz<br>— High-speed arithmetic operations<br>8/16/32-bit register-register add/subtract : 50 ns<br>16 × 16-bit register-register multiply : 200 ns<br>16 × 16 + 42-bit multiply and accumulate : 200 ns<br>32 ÷ 16-bit register-register divide : 1000 ns<br>• Instruction set suitable for high-speed operation<br>— Sixty-nine basic instructions<br>— 8/16/32-bit move/arithmetic and logic instructions<br>— Unsigned/signed multiply and divide instructions<br>— Multiply-and accumulate instruction<br>— Powerful bit-manipulation instructions<br>• Two CPU operating modes<br>— Normal mode : 64-kbyte address space<br>— Advanced mode : 16-Mbyte address space |
| Bus controller | • Address space divided into 8 areas, with bus specifications settable independently for each area<br>• Chip select output possible for each area<br>• Choice of 8-bit or 16-bit access space for each area<br>• 2-state or 3-state access space can be designated for each area<br>• Number of program wait states can be set for each area<br>• Burst ROM directly connectable<br>• Maximum 8-Mbyte DRAM or PSRAM directly connectable (or use of interval timer possible)<br>• External bus release function |
| DMA controller (DMAC) | • Choice of short address mode or full address mode<br>• 4 channels in short address mode<br>• 2 channels in full address mode<br>• Transfer possible in repeat mode, block transfer mode, etc.<br>• Single address mode transfer possible<br>• Can be activated by internal interrupt |

**Table 1-1 Overview (cont)**

| Item | Specification |
|---|---|
| Data transfer controller (DTC) | • Can be activated by internal interrupt or software<br>• Multiple transfers or multiple types of transfer possible for one activation source<br>• Transfer possible in repeat mode, block transfer mode, etc.<br>• Request can be sent to CPU for interrupt that activated DTC |
| 16-bit timer-pulse unit (TPU) | • 6-channel 16-bit timer on-chip<br>• Pulse I/O processing capability for up to 16 pins'<br>• Automatic 2-phase encoder count capability |
| Programmable pulse generator (PPG) | • Maximum 16-bit pulse output possible with TPU as time base<br>• Output trigger selectable in 4-bit groups<br>• Non-overlap margin can be set<br>• Direct output or inverse output setting possible |
| 8-bit timer 2 channels | • 8-bit up-counter (external event count capability)<br>• Two time constant registers<br>• Two-channel connection possible |
| Watchdog timer | • Watchdog timer or interval timer selectable |
| Serial communication interface (SCI) 3 channels | • Asynchronous mode or synchronous mode selectable<br>• Multiprocessor communication function<br>• Smart card interface function |
| A/D converter | • Resolution: 10 bits<br>• Input: 8 channels<br>• High-speed conversion : 2.2 µs minimum conversion time (at 20 MHz operation)<br>• Single or scan mode selectable<br>• Sample and hold circuit<br>• A/D conversion can be activated by external trigger or timer trigger |
| D/A converter | • Resolution: 8 bits<br>• Output: 2 channels |
| I/O ports | • 87 I/O pins, 8 input-only pins |
| Memory | • PROM or mask ROM<br>• High-speed static RAM |
| Interrupt controller | • Nine external interrupt pins (NMI, $\overline{IRQ}_0$ to $\overline{IRQ}_7$)<br>• 52 internal interrupt sources<br>• Eight priority levels settable |

| Product Name | ROM | RAM |
|---|---|---|
| H8S/2655 | 128 kbytes | 4 kbytes |
| H8S/2653 | 64 kbytes | 4 kbytes |

**Table 1-1 Overview (cont)**

| Item | Specification |
|---|---|
| Power-down state | • Medium-speed mode<br>• Sleep mode<br>• Module stop mode<br>• Software standby mode<br>• Hardware standby mode |
| Operating modes | Seven MCU operating modes |

| Mode | CPU Operating Mode | Description | On-Chip ROM | External Data Bus Initial Value | External Data Bus Maximum Value |
|---|---|---|---|---|---|
| 1 | Normal | On-chip ROM disabled expansion mode | Disabled | 8 bits | 16 bits |
| 2 | | On-chip ROM enabled expansion mode | Enabled | 8 bits | 16 bits |
| 3 | | Single-chip mode | Enabled | — | |
| 4 | Advanced | On-chip ROM disabled expansion mode | Disabled | 16 bits | 16 bits |
| 5 | | On-chip ROM disabled expansion mode | Disabled | 8 bits | 16 bits |
| 6 | | On-chip ROM enabled expansion mode | Enabled | 8 bits | 16 bits |
| 7 | | Single-chip mode | Enabled | — | |

| | |
|---|---|
| Clock pulse generator | • Built-in duty correction circuit |
| Packages | • 120-pin plastic TQFP (TFP-120)<br>• 128-pin plastic QFP (FP-128) |

Product lineup

| Model Name | | | |
|---|---|---|---|
| 5 V Version ($V_{cc}$ = 5 V ± 10%) | Low-Voltage Version ($V_{cc}$ = 2.7 to 5.5 V) | ROM | Packages |
| HD6472655TE | HD6472655VTE | PROM | TFP-120 |
| HD6472655F | HD6472655VF | | FP-128 |
| HD6432655(***)TE | HD6432655(***)TE | Mask ROM | TFP-120 |
| HD6432655(***)F | HD6432655(***)F | | FP-128 |
| HD6432653(***)TE | HD6432653(***)TE | | TFP-120 |
| HD6432653(***)F | HD6432653(***)F | | FP-128 |

Legend: Marked (***) is ROM code.

## 1.2 Block Diagram

Figure 1-1 shows an internal block diagram of the H8S/2655 Series.

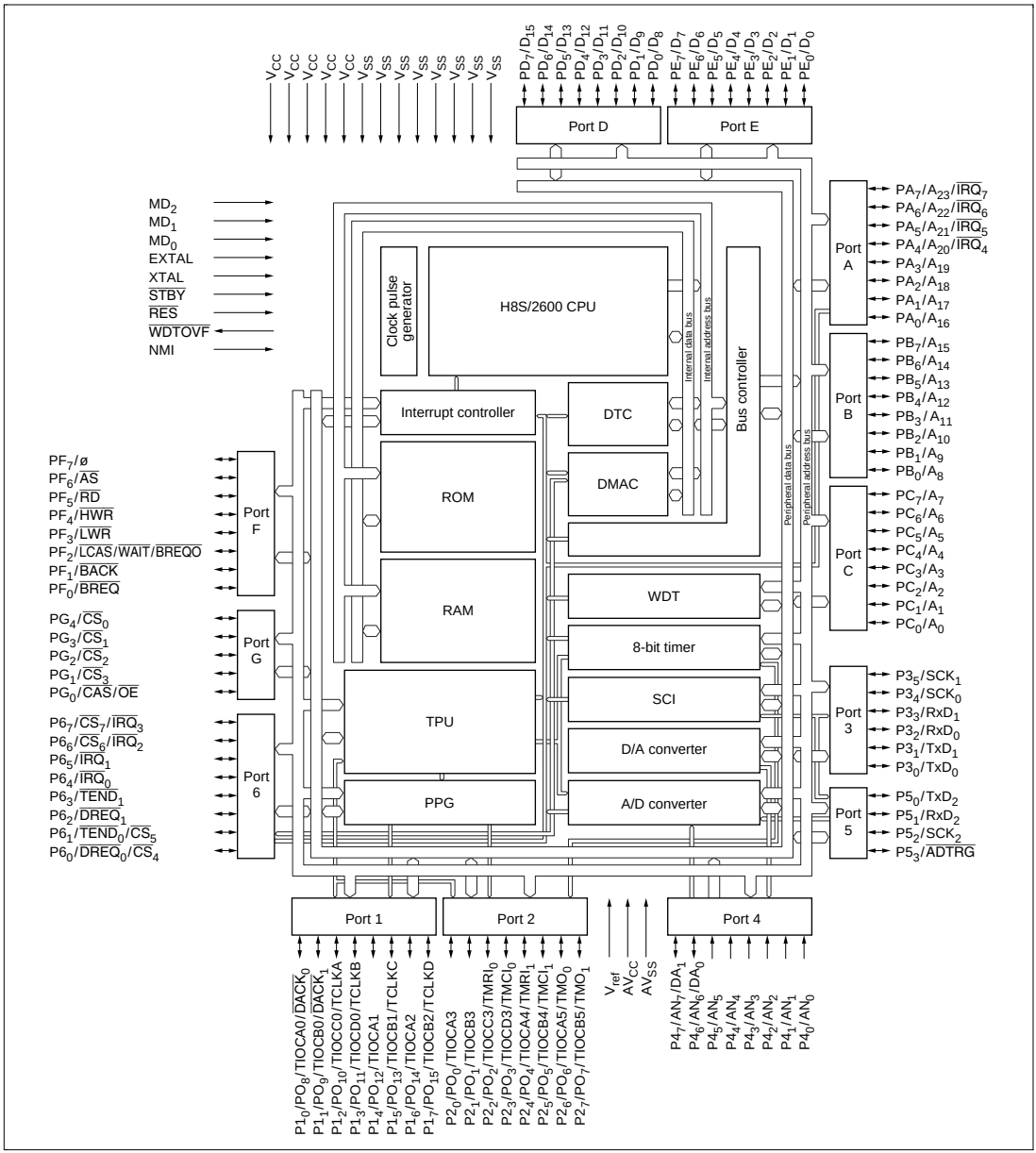



**Figure 1-1 Block Diagram**

## 1.3 Pin Description

### 1.3.1 Pin Arrangement

Figures 1-2 and 1-3 show the pin arrangement of the H8S/2655 Series.

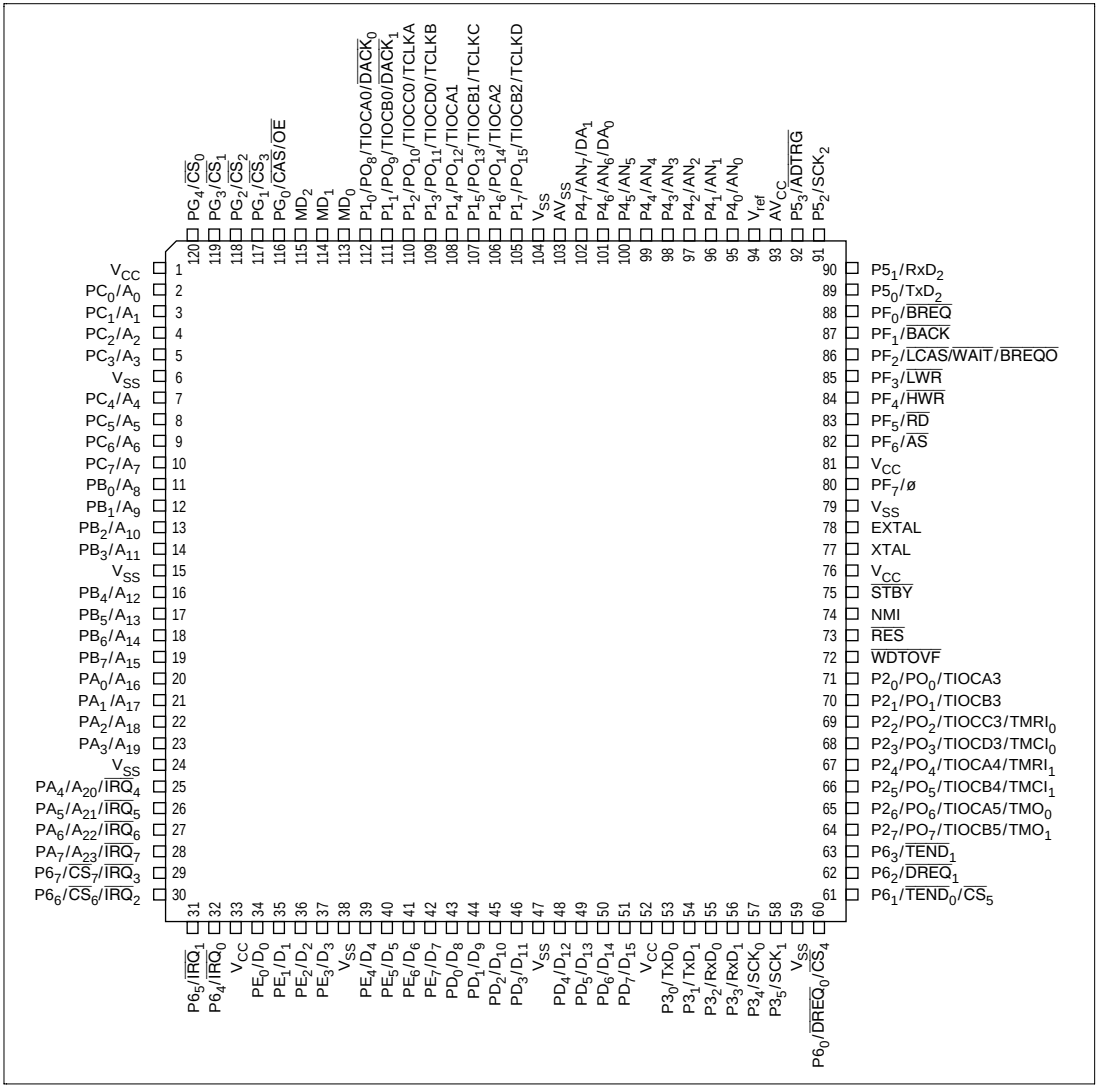



**Figure 1-2 Pin Arrangement (TFP-120: Top View)**

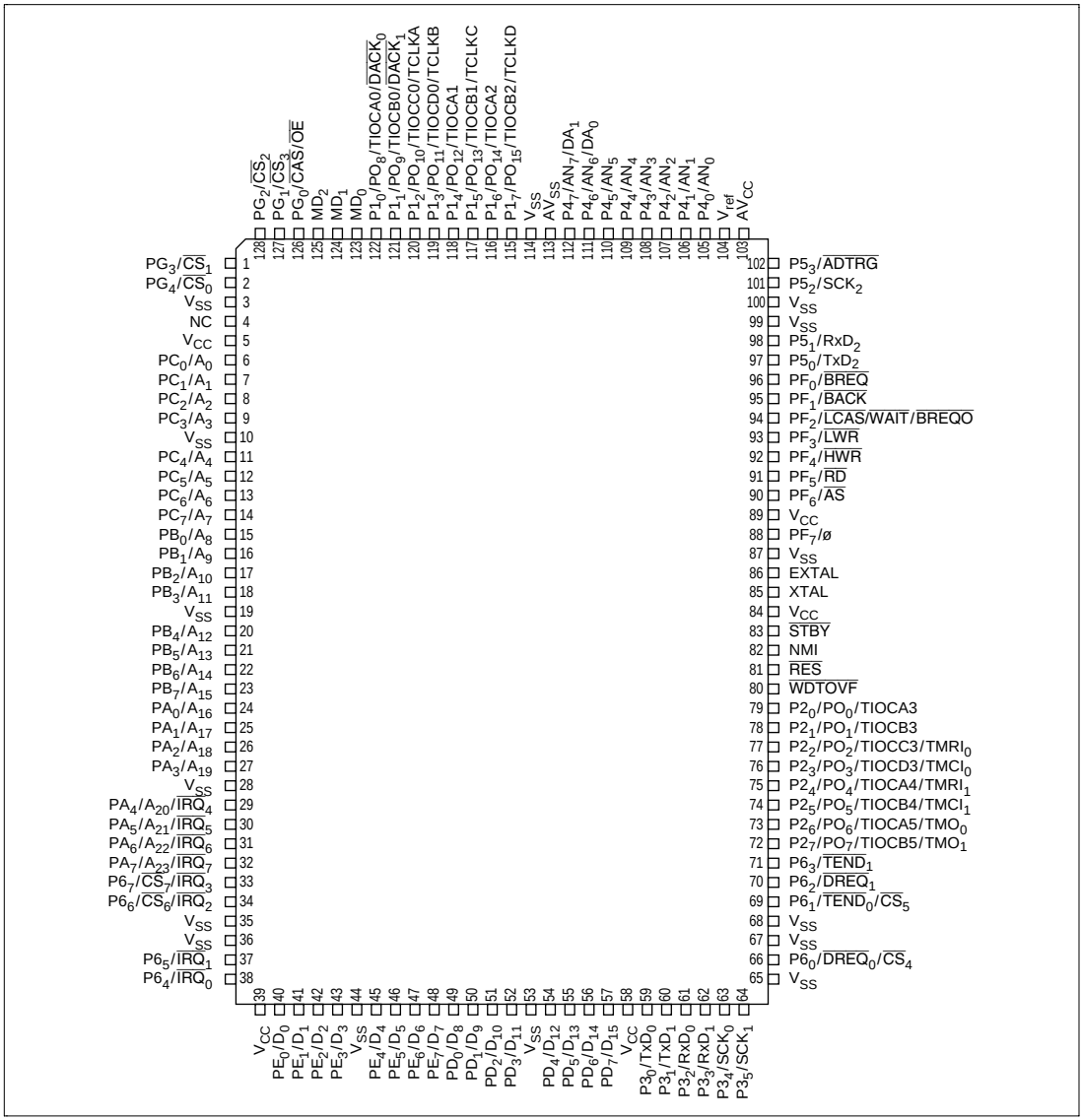



**Figure 1-3 Pin Arrangement (FP-128: Top View)**

### 1.3.2 Pin Functions in Each Operating Mode

Table 1-2 shows the pin functions of the H8S/2655 Series in each of the operating modes.

**Table 1-2 Pin Functions in Each Operating Mode**

| Pin No. | | Pin Name | | | | | | | PROM |
|---|---|---|---|---|---|---|---|---|---|
| TFP-120 | FP-128 | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 | Mode |
| 1 | 5 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 2 | 6 | $A_0$ | $PC_0/A_0$ | $PC_0$ | $A_0$ | $A_0$ | $PC_0/A_0$ | $PC_0$ | $A_0$ |
| 3 | 7 | $A_1$ | $PC_1/A_1$ | $PC_1$ | $A_1$ | $A_1$ | $PC_1/A_1$ | $PC_1$ | $A_1$ |
| 4 | 8 | $A_2$ | $PC_2/A_2$ | $PC_2$ | $A_2$ | $A_2$ | $PC_2/A_2$ | $PC_2$ | $A_2$ |
| 5 | 9 | $A_3$ | $PC_3/A_3$ | $PC_3$ | $A_3$ | $A_3$ | $PC_3/A_3$ | $PC_3$ | $A_3$ |
| 6 | 10 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 7 | 11 | $A_4$ | $PC_4/A_4$ | $PC_4$ | $A_4$ | $A_4$ | $PC_4/A_4$ | $PC_4$ | $A_4$ |
| 8 | 12 | $A_5$ | $PC_5/A_5$ | $PC_5$ | $A_5$ | $A_5$ | $PC_5/A_5$ | $PC_5$ | $A_5$ |
| 9 | 13 | $A_6$ | $PC_6/A_6$ | $PC_6$ | $A_6$ | $A_6$ | $PC_6/A_6$ | $PC_6$ | $A_6$ |
| 10 | 14 | $A_7$ | $PC_7/A_7$ | $PC_7$ | $A_7$ | $A_7$ | $PC_7/A_7$ | $PC_7$ | $A_7$ |
| 11 | 15 | $A_8$ | $PB_0/A_8$ | $PB_0$ | $A_8$ | $A_8$ | $PB_0/A_8$ | $PB_0$ | $A_8$ |
| 12 | 16 | $A_9$ | $PB_1/A_9$ | $PB_1$ | $A_9$ | $A_9$ | $PB_1/A_9$ | $PB_1$ | $\overline{OE}$ |
| 13 | 17 | $A_{10}$ | $PB_2/A_{10}$ | $PB_2$ | $A_{10}$ | $A_{10}$ | $PB_2/A_{10}$ | $PB_2$ | $A_{10}$ |
| 14 | 18 | $A_{11}$ | $PB_3/A_{11}$ | $PB_3$ | $A_{11}$ | $A_{11}$ | $PB_3/A_{11}$ | $PB_3$ | $A_{11}$ |
| 15 | 19 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 16 | 20 | $A_{12}$ | $PB_4/A_{12}$ | $PB_4$ | $A_{12}$ | $A_{12}$ | $PB_4/A_{12}$ | $PB_4$ | $A_{12}$ |
| 17 | 21 | $A_{13}$ | $PB_5/A_{13}$ | $PB_5$ | $A_{13}$ | $A_{13}$ | $PB_5/A_{13}$ | $PB_5$ | $A_{13}$ |
| 18 | 22 | $A_{14}$ | $PB_6/A_{14}$ | $PB_6$ | $A_{14}$ | $A_{14}$ | $PB_6/A_{14}$ | $PB_6$ | $A_{14}$ |
| 19 | 23 | $A_{15}$ | $PB_7/A_{15}$ | $PB_7$ | $A_{15}$ | $A_{15}$ | $PB_7/A_{15}$ | $PB_7$ | $A_{15}$ |
| 20 | 24 | $PA_0$ | $PA_0$ | $PA_0$ | $A_{16}$ | $A_{16}$ | $PA_0/A_{16}$ | $PA_0$ | $A_{16}$ |
| 21 | 25 | $PA_1$ | $PA_1$ | $PA_1$ | $A_{17}$ | $A_{17}$ | $PA_1/A_{17}$ | $PA_1$ | $V_{CC}$ |
| 22 | 26 | $PA_2$ | $PA_2$ | $PA_2$ | $A_{18}$ | $A_{18}$ | $PA_2/A_{18}$ | $PA_2$ | $V_{CC}$ |
| 23 | 27 | $PA_3$ | $PA_3$ | $PA_3$ | $A_{19}$ | $A_{19}$ | $PA_3/A_{19}$ | $PA_3$ | NC |
| 24 | 28 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 25 | 29 | $PA_4/\overline{IRQ_4}$ | $PA_4/\overline{IRQ_4}$ | $PA_4/\overline{IRQ_4}$ | $A_{20}$ | $A_{20}$ | $PA_4/A_{20}/\overline{IRQ_4}$ | $PA_4/\overline{IRQ_4}$ | NC |
| 26 | 30 | $PA_5/\overline{IRQ_5}$ | $PA_5/\overline{IRQ_5}$ | $PA_5/\overline{IRQ_5}$ | $PA_5/A_{21}/\overline{IRQ_5}$ | $PA_5/A_{21}/\overline{IRQ_5}$ | $PA_5/A_{21}/\overline{IRQ_5}$ | $PA_5/\overline{IRQ_5}$ | NC |
| 27 | 31 | $PA_6/\overline{IRQ_6}$ | $PA_6/\overline{IRQ_6}$ | $PA_6/\overline{IRQ_6}$ | $PA_6/A_{22}/\overline{IRQ_6}$ | $PA_6/A_{22}/\overline{IRQ_6}$ | $PA_6/A_{22}/\overline{IRQ_6}$ | $PA_6/\overline{IRQ_6}$ | NC |

**Table 1-2 Pin Functions in Each Operating Mode (cont)**

| Pin No. | | Pin Name | | | | | | | PROM |
|---|---|---|---|---|---|---|---|---|---|
| TFP-120 | FP-128 | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 | Mode |
| 28 | 32 | $PA_7/\overline{IRQ_7}$ | $PA_7/\overline{IRQ_7}$ | $PA_7/\overline{IRQ_7}$ | $PA_7/A_{23}/\overline{IRQ_7}$ | $PA_7/A_{23}/\overline{IRQ_7}$ | $PA_7/A_{23}/\overline{IRQ_7}$ | $PA_7/\overline{IRQ_7}$ | NC |
| 29 | 33 | $P6_7/\overline{IRQ_3}$ | $P6_7/\overline{IRQ_3}$ | $P6_7/\overline{IRQ_3}$ | $P6_7/\overline{IRQ_3}/\overline{CS_7}$ | $P6_7/\overline{IRQ_3}/\overline{CS_7}$ | $P6_7/\overline{IRQ_3}/\overline{CS_7}$ | $P6_7/\overline{IRQ_3}$ | NC |
| 30 | 34 | $P6_6/\overline{IRQ_2}$ | $P6_6/\overline{IRQ_2}$ | $P6_6/\overline{IRQ_2}$ | $P6_6/\overline{IRQ_2}/\overline{CS_6}$ | $P6_6/\overline{IRQ_2}/\overline{CS_6}$ | $P6_6/\overline{IRQ_2}/\overline{CS_6}$ | $P6_6/\overline{IRQ_2}$ | NC |
| — | 35 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| — | 36 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 31 | 37 | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | $P6_5/\overline{IRQ_1}$ | NC |
| 32 | 38 | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | $P6_4/\overline{IRQ_0}$ | NC |
| 33 | 39 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 34 | 40 | $PE_0/D_0$ | $PE_0/D_0$ | $PE_0$ | $PE_0/D_0$ | $PE_0/D_0$ | $PE_0/D_0$ | $PE_0$ | NC |
| 35 | 41 | $PE_1/D_1$ | $PE_1/D_1$ | $PE_1$ | $PE_1/D_1$ | $PE_1/D_1$ | $PE_1/D_1$ | $PE_1$ | NC |
| 36 | 42 | $PE_2/D_2$ | $PE_2/D_2$ | $PE_2$ | $PE_2/D_2$ | $PE_2/D_2$ | $PE_2/D_2$ | $PE_2$ | NC |
| 37 | 43 | $PE_3/D_3$ | $PE_3/D_3$ | $PE_3$ | $PE_3/D_3$ | $PE_3/D_3$ | $PE_3/D_3$ | $PE_3$ | NC |
| 38 | 44 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 39 | 45 | $PE_4/D_4$ | $PE_4/D_4$ | $PE_4$ | $PE_4/D_4$ | $PE_4/D_4$ | $PE_4/D_4$ | $PE_4$ | NC |
| 40 | 46 | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5$ | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5$ | NC |
| 41 | 47 | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6$ | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6$ | NC |
| 42 | 48 | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7$ | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7$ | NC |
| 43 | 49 | $D_8$ | $D_8$ | $PD_0$ | $D_8$ | $D_8$ | $D_8$ | $PD_0$ | $D_0$ |
| 44 | 50 | $D_9$ | $D_9$ | $PD_1$ | $D_9$ | $D_9$ | $D_9$ | $PD_1$ | $D_1$ |
| 45 | 51 | $D_{10}$ | $D_{10}$ | $PD_2$ | $D_{10}$ | $D_{10}$ | $D_{10}$ | $PD_2$ | $D_2$ |
| 46 | 52 | $D_{11}$ | $D_{11}$ | $PD_3$ | $D_{11}$ | $D_{11}$ | $D_{11}$ | $PD_3$ | $D_3$ |
| 47 | 53 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 48 | 54 | $D_{12}$ | $D_{12}$ | $PD_4$ | $D_{12}$ | $D_{12}$ | $D_{12}$ | $PD_4$ | $D_4$ |
| 49 | 55 | $D_{13}$ | $D_{13}$ | $PD_5$ | $D_{13}$ | $D_{13}$ | $D_{13}$ | $PD_5$ | $D_5$ |
| 50 | 56 | $D_{14}$ | $D_{14}$ | $PD_6$ | $D_{14}$ | $D_{14}$ | $D_{14}$ | $PD_6$ | $D_6$ |
| 51 | 57 | $D_{15}$ | $D_{15}$ | $PD_7$ | $D_{15}$ | $D_{15}$ | $D_{15}$ | $PD_7$ | $D_7$ |
| 52 | 58 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 53 | 59 | $P3_0/TxD_0$ | $P3_0/TxD_0$ | $P3_0/TxD_0$ | $P3_0/TxD_0$ | $P3_0/TxD_0$ | $P3_0/TxD_0$ | $P3_0/TxD_0$ | NC |
| 54 | 60 | $P3_1/TxD_1$ | $P3_1/TxD_1$ | $P3_1/TxD_1$ | $P3_1/TxD_1$ | $P3_1/TxD_1$ | $P3_1/TxD_1$ | $P3_1/TxD_1$ | NC |
| 55 | 61 | $P3_2/RxD_0$ | $P3_2/RxD_0$ | $P3_2/RxD_0$ | $P3_2/RxD_0$ | $P3_2/RxD_0$ | $P3_2/RxD_0$ | $P3_2/RxD_0$ | NC |

**Table 1-2 Pin Functions in Each Operating Mode (cont)**

| Pin No. | | Pin Name | | | | | | | PROM |
|---|---|---|---|---|---|---|---|---|---|
| TFP-120 | FP-128 | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 | Mode |
| 56 | 62 | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | $P3_3$/$RxD_1$ | NC |
| 57 | 63 | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | $P3_4$/$SCK_0$ | NC |
| 58 | 64 | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | $P3_5$/$SCK_1$ | NC |
| 59 | 65 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 60 | 66 | $P6_0$/$\overline{DREQ_0}$ | $P6_0$/$\overline{DREQ_0}$ | $P6_0$/$\overline{DREQ_0}$ | $P6_0$/$\overline{DREQ_0}$/$\overline{CS_4}$ | $P6_0$/$\overline{DREQ_0}$/$\overline{CS_4}$ | $P6_0$/$\overline{DREQ_0}$/$\overline{CS_4}$ | $P6_0$/$\overline{DREQ_0}$ | NC |
| — | 67 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| — | 68 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 61 | 69 | $P6_1$/$\overline{TEND_0}$ | $P6_1$/$\overline{TEND_0}$ | $P6_1$/$\overline{TEND_0}$ | $P6_1$/$\overline{TEND_0}$/$\overline{CS_5}$ | $P6_1$/$\overline{TEND_0}$/$\overline{CS_5}$ | $P6_1$/$\overline{TEND_0}$/$\overline{CS_5}$ | $P6_1$/$\overline{TEND_0}$ | NC |
| 62 | 70 | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | $P6_2$/$\overline{DREQ_1}$ | NC |
| 63 | 71 | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | $P6_3$/$\overline{TEND_1}$ | NC |
| 64 | 72 | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | $P2_7$/$PO_7$/TIOCB5/$TMO_1$ | NC |
| 65 | 73 | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | $P2_6$/$PO_6$/TIOCA5/$TMO_0$ | NC |
| 66 | 74 | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | $P2_5$/$PO_5$/TIOCB4/$TMCI_1$ | NC |
| 67 | 75 | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | $P2_4$/$PO_4$/TIOCA4/$TMRI_1$ | NC |
| 68 | 76 | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | $P2_3$/$PO_3$/TIOCD3/$TMCI_0$ | NC |
| 69 | 77 | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | $P2_2$/$PO_2$/TIOCC3/$TMRI_1$ | NC |
| 70 | 78 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | $P2_1$/$PO_1$/TIOCB3 | NC |
| 71 | 79 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | $P2_0$/$PO_0$/TIOCA3 | NC |
| 72 | 80 | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | NC |

**Table 1-2 Pin Functions in Each Operating Mode (cont)**

| Pin No. | | Pin Name | | | | | | | PROM |
|---|---|---|---|---|---|---|---|---|---|
| TFP-120 | FP-128 | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 | Mode |
| 73 | 81 | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $V_{PP}$ |
| 74 | 82 | NMI | NMI | NMI | NMI | NMI | NMI | NMI | $A_9$ |
| 75 | 83 | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $V_{SS}$ |
| 76 | 84 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 77 | 85 | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL | NC |
| 78 | 86 | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | NC |
| 79 | 87 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 80 | 88 | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | NC |
| 81 | 89 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 82 | 90 | $\overline{AS}$ | $\overline{AS}$ | $PF_6$ | $\overline{AS}$ | $\overline{AS}$ | $\overline{AS}$ | $PF_6$ | NC |
| 83 | 91 | $\overline{RD}$ | $\overline{RD}$ | $PF_5$ | $\overline{RD}$ | $\overline{RD}$ | $\overline{RD}$ | $PF_5$ | NC |
| 84 | 92 | $\overline{HWR}$ | $\overline{HWR}$ | $PF_4$ | $\overline{HWR}$ | $\overline{HWR}$ | $\overline{HWR}$ | $PF_4$ | NC |
| 85 | 93 | $\overline{LWR}$ | $\overline{LWR}$ | $PF_3$ | $\overline{LWR}$ | $\overline{LWR}$ | $\overline{LWR}$ | $PF_3$ | NC |
| 86 | 94 | $PF_2$/$\overline{WAIT}$/ $\overline{BREQO}$ | $PF_2$/$\overline{WAIT}$/ $\overline{BREQO}$ | $PF_2$ | $PF_2$/$\overline{LCAS}$/ $\overline{WAIT}$/ $\overline{BREQO}$ | $PF_2$/$\overline{LCAS}$/ $\overline{WAIT}$/ $\overline{BREQO}$ | $PF_2$/$\overline{LCAS}$/ $\overline{WAIT}$/ $\overline{BREQO}$ | $PF_2$ | $\overline{CE}$ |
| 87 | 95 | $PF_1$/$\overline{BACK}$ | $PF_1$/$\overline{BACK}$ | $PF_1$ | $PF_1$/$\overline{BACK}$ | $PF_1$/$\overline{BACK}$ | $PF_1$/$\overline{BACK}$ | $PF_1$ | $\overline{PGM}$ |
| 88 | 96 | $PF_0$/$\overline{BREQ}$ | $PF_0$/$\overline{BREQ}$ | $PF_0$ | $PF_0$/$\overline{BREQ}$ | $PF_0$/$\overline{BREQ}$ | $PF_0$/$\overline{BREQ}$ | $PF_0$ | NC |
| 89 | 97 | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | $P5_0$/$TxD_2$ | NC |
| 90 | 98 | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | $P5_1$/$RxD_2$ | NC |
| — | 99 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| — | 100 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 91 | 101 | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | $P5_2$/$SCK_2$ | NC |
| 92 | 102 | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | $P5_3$/ $\overline{ADTRG}$ | NC |
| 93 | 103 | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $V_{CC}$ |
| 94 | 104 | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{CC}$ |
| 95 | 105 | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | $P4_0$/$AN_0$ | NC |
| 96 | 106 | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | $P4_1$/$AN_1$ | NC |
| 97 | 107 | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | $P4_2$/$AN_2$ | NC |
| 98 | 108 | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | $P4_3$/$AN_3$ | NC |
| 99 | 109 | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | $P4_4$/$AN_4$ | NC |
| 100 | 110 | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | $P4_5$/$AN_5$ | NC |
| 101 | 111 | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | $P4_6$/$AN_6$/ $DA_0$ | NC |

**Table 1-2 Pin Functions in Each Operating Mode (cont)**

| Pin No. | | Pin Name | | | | | | | PROM |
|---|---|---|---|---|---|---|---|---|---|
| TFP-120 | FP-128 | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 | Mode |
| 102 | 112 | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | $P4_7$/$AN_7$/ $DA_1$ | NC |
| 103 | 113 | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $V_{SS}$ |
| 104 | 114 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 105 | 115 | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | $P1_7$/$PO_{15}$/ TIOCB2/ TCLKD | NC |
| 106 | 116 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | $P1_6$/$PO_{14}$/ TIOCA2 | NC |
| 107 | 117 | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | $P1_5$/$PO_{13}$/ TIOCB1/ TCLKC | NC |
| 108 | 118 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | $P1_4$/$PO_{12}$/ TIOCA1 | NC |
| 109 | 119 | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | $P1_3$/$PO_{11}$/ TIOCD0/ TCLKB | NC |
| 110 | 120 | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | $P1_2$/$PO_{10}$/ TIOCC0/ TCLKA | NC |
| 111 | 121 | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | $P1_1$/$PO_9$/ TIOCB0/ $\overline{DACK_1}$ | NC |
| 112 | 122 | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | $P1_0$/$PO_8$/ TIOCA0/ $\overline{DACK_0}$ | NC |
| 113 | 123 | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $V_{SS}$ |
| 114 | 124 | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $V_{SS}$ |
| 115 | 125 | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $V_{SS}$ |
| 116 | 126 | $PG_0$ | $PG_0$ | $PG_0$ | $PG_0$/$\overline{CAS}$/ $\overline{OE}$ | $PG_0$/$\overline{CAS}$/ $\overline{OE}$ | $PG_0$/$\overline{CAS}$/ $\overline{OE}$ | $PG_0$ | NC |
| 117 | 127 | $PG_1$ | $PG_1$ | $PG_1$ | $PG_1$/$\overline{CS_3}$ | $PG_1$/$\overline{CS_3}$ | $PG_1$/$\overline{CS_3}$ | $PG_1$ | NC |
| 118 | 128 | $PG_2$ | $PG_2$ | $PG_2$ | $PG_2$/$\overline{CS_2}$ | $PG_2$/$\overline{CS_2}$ | $PG_2$/$\overline{CS_2}$ | $PG_2$ | NC |
| 119 | 1 | $PG_3$ | $PG_3$ | $PG_3$ | $PG_3$/$\overline{CS_1}$ | $PG_3$/$\overline{CS_1}$ | $PG_3$/$\overline{CS_1}$ | $PG_3$ | NC |
| 120 | 2 | $PG_4$/$\overline{CS_0}$ | $PG_4$/$\overline{CS_0}$ | $PG_4$ | $PG_4$/$\overline{CS_0}$ | $PG_4$/$\overline{CS_0}$ | $PG_4$/$\overline{CS_0}$ | $PG_4$ | NC |
| — | 3 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| — | 4 | NC | NC | NC | NC | NC | NC | NC | NC |

Note: NC pins should be connected to $V_{SS}$ or left open.

### 1.3.3 Pin Functions

Table 1-3 outlines the pin functions of the H8S/2655 Series.

**Table 1-3 Pin Functions**

| | | Pin No. | | | |
|---|---|---|---|---|---|
| **Type** | **Symbol** | **TFP-120** | **FP-128** | **I/O** | **Name and Function** |
| Power | $V_{CC}$ | 1, 33, 52, 76, 81 | 5, 39, 58, 84, 89 | Input | Power supply: For connection to the power supply. All $V_{CC}$ pins should be connected to the system power supply. |
| | $V_{SS}$ | 6, 15, 24, 38, 47, 59, 79, 104 | 3, 10, 19, 28, 35, 36, 44, 53, 65, 67, 68, 87, 99, 100, 114 | Input | Ground: For connection to ground (0 V). All $V_{SS}$ pins should be connected to the system power supply (0 V). |
| Clock | XTAL | 77 | 85 | Input | Connects to a crystal oscillator. See section 20, Clock Pulse Generator, for typical connection diagrams for a crystal oscillator and external clock input. |
| | EXTAL | 78 | 86 | Input | Connects to a crystal oscillator. The EXTAL pin can also input an external clock. See section 20, Clock Pulse Generator, for typical connection diagrams for a crystal oscillator and external clock input. |
| | ø | 80 | 88 | Output | System clock: Supplies the system clock to an external device. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| Operating mode control | $MD_2$ to $MD_0$ | 115 to 113 | 125 to 123 | Input | Mode pins: These pins set the operating mode.<br>The relation between the settings of pins $MD_2$ to $MD_0$ and the operating mode is shown below. These pins should not be changed while the H8S/2655 Series is operating. |

| MD2 | MD1 | MD0 | Operating Mode |
|---|---|---|---|
| 0 | 0 | 0 | — |
| | | 1 | Mode 1 |
| | 1 | 0 | Mode 2 |
| | | 1 | Mode 3 |
| 1 | 0 | 0 | Mode 4 |
| | | 1 | Mode 5 |
| | 1 | 0 | Mode 6 |
| | | 1 | Mode 7 |

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| System control | $\overline{RES}$ | 73 | 81 | Input | Reset input: When this pin is driven low, the chip is reset. The type of reset can be selected according to the NMI input level. At power-on, the NMI pin input level should be set high. |
| | $\overline{STBY}$ | 75 | 83 | Input | Standby: When this pin is driven low, a transition is made to hardware standby mode. |
| | $\overline{BREQ}$ | 88 | 96 | Input | Bus request: Used by an external bus master to issue a bus request to the H8S/2655 Series. |
| | $\overline{BREQO}$ | 86 | 94 | Output | Bus request output: The external bus request signal used when an internal bus master accesses external space in the external bus-released state. |
| | $\overline{BACK}$ | 87 | 95 | Output | Bus request acknowledge: Indicates that the bus has been released to an external bus master. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| Interrupts | NMI | 74 | 82 | Input | Nonmaskable interrupt: Requests a nonmaskable interrupt. When this pin is not used, it should be fixed high. |
| | $\overline{IRQ}_7$ to $\overline{IRQ}_0$ | 28 to 25, 29 to 32 | 32 to 29, 33, 34, 37, 38 | Input | Interrupt request 7 to 0: These pins request a maskable interrupt. |
| Address bus | $A_{23}$ to $A_0$ | 28 to 25, 23 to 16, 14 to 7, 5 to 2 | 32 to 29, 27 to 20, 18 to 11, 9 to 6 | Output | Address bus: These pins output an address. |
| Data bus | $D_{15}$ to $D_0$ | 51 to 48, 46 to 39, 37 to 34 | 57 to 54, 52 to 45, 43 to 40 | I/O | Data bus: These pins constitute a bidirectional data bus. |
| Bus control | $\overline{CS}_7$ to $\overline{CS}_0$ | 29, 30, 61, 60, 117 to 120 | 33, 34, 69, 66, 127, 128, 1, 2 | Output | Chip select: Signals for selecting areas 7 to 0. |
| | $\overline{AS}$ | 82 | 90 | Output | Address strobe: When this pin is low, it indicates that address output on the address bus is enabled. |
| | $\overline{RD}$ | 83 | 91 | Output | Read: When this pin is low, it indicates that the external address space can be read. |
| | $\overline{HWR}$ | 84 | 92 | Output | High write/write enable/upper write enable:<br>A strobe signal that writes to external space and indicates that the upper half ($D_{15}$ to $D_8$) of the data bus is enabled.<br>The 2CAS type DRAM write enable signal.<br>The 2WE type DRAM upper write enable signal. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| Bus control | $\overline{\text{LWR}}$ | 85 | 93 | Output | Low write/lower column address strobe/lower write enable:<br>A strobe signal that writes to external space and indicates that the lower half ($D_7$ to $D_0$) of the data bus is enabled.<br>The 2CAS type (LCASS = 1) DRAM lower column address strobe signal.<br>The 2WE type DRAM lower write enable signal. |
| | $\overline{\text{CAS}}$/ $\overline{\text{OE}}$ | 116 | 126 | Output | Upper column address strobe/column address strobe/output enable/refresh:<br>The 2CAS type DRAM upper column address strobe signal.<br>The 2WE type DRAM column address strobe signal.<br>The PSRAM output enable signal. |
| | $\overline{\text{WAIT}}$ | 86 | 94 | Input | Wait: Requests insertion of a wait state in the bus cycle when accessing external 3-state address space. |
| | $\overline{\text{LCAS}}$ | 86 | 94 | Output | Lower column address strobe: The 2-CAS type (LCASS = 0) DRAM lower column address strobe signal |
| DMA controller (DMAC) | $\overline{\text{DREQ}}_1$, $\overline{\text{DREQ}}_0$ | 62, 60 | 70, 66 | Input | DMA request 1 and 0: These pins request DMAC activation. |
| | $\overline{\text{TEND}}_1$, $\overline{\text{TEND}}_0$ | 63, 61 | 71, 69 | Output | DMA transfer end 1 and 0: These pins indicate the end of DMAC data transfer. |
| | $\overline{\text{DACK}}_1$, $\overline{\text{DACK}}_0$ | 111, 112 | 121, 122 | Output | DMA transfer acknowledge 1 and 0: These are the DMAC single address transfer acknowledge pins. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| 16-bit timer-pulse unit (TPU) | TCLKD to TCLKA | 105, 107, 109, 110 | 115, 117, 119, 120 | Input | Clock input D to A: These pins input an external clock. |
| | TIOCA0, TIOCB0, TIOCC0, TIOCD0 | 112 to 109 | 122 to 119 | I/O | Input capture/ output compare match A0 to D0: The TGR0A to TGR0D input capture input or output compare output, or PWM output pins. |
| | TIOCA1, TIOCB1 | 108, 107 | 118, 117 | I/O | Input capture/ output compare match A1 and B1: The TGR1A and TGR1B input capture input or output compare output, or PWM output pins. |
| | TIOCA2, TIOCB2 | 106, 105 | 116, 115 | I/O | Input capture/ output compare match A2 and B2: The TGR2A and TGR2B input capture input or output compare output, or PWM output pins. |
| | TIOCA3, TIOCB3, TIOCC3, TIOCD3 | 71 to 68 | 79 to 76 | I/O | Input capture/ output compare match A3 to D3: The TGR3A to TGR3D input capture input or output compare output, or PWM output pins. |
| | TIOCA4, TIOCB4 | 67, 66 | 75, 74 | I/O | Input capture/ output compare match A4 and B4: The TGR4A and TGR4B input capture input or output compare output, or PWM output pins. |
| | TIOCA5, TIOCB5 | 65, 64 | 73, 72 | I/O | Input capture/ output compare match A5 and B5: The TGR5A and TGR5B input capture input or output compare output, or PWM output pins. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| Programmable pulse generator (PPG) | $PO_{15}$ to $PO_0$ | 105 to 112, 64 to 71 | 115 to 122, 72 to 79 | Output | Pulse output 15 to 0: Pulse output pins. |
| 8-bit timer | $TMO_0$, $TMO_1$ | 65, 64 | 73, 72 | Output | Compare match output: The compare match output pins. |
| | $TMCI_0$, $TMCI_1$ | 68, 66 | 76, 74 | Input | Counter external clock input: Input pins for the external clock input to the counter. |
| | $TMRI_0$, $TMRI_1$ | 69, 67 | 77, 75 | Input | Counter external reset input: The counter reset input pins. |
| Watchdog timer (WDT) | $\overline{WDTOVF}$ | 72 | 80 | Output | Watchdog timer overflows: The counter overflows signal output pin in watchdog timer mode. |
| Serial communication interface (SCI) Smart Card interface | $TxD_2$, $TxD_1$, $TxD_0$ | 89, 54, 53 | 97, 60, 59 | Output | Transmit data (channel 0, 1, 2): Data output pins. |
| | $RxD_2$, $RxD_1$, $RxD_0$ | 90, 56, 55 | 98, 62, 61 | Input | Receive data (channel 0, 1, 2): Data input pins. |
| | $SCK_2$, $SCK_1$ $SCK_0$ | 91, 58 57 | 101, 64, 63 | I/O | Serial clock (channel 0, 1, 2): Clock I/O pins. |
| A/D converter | $AN_7$ to $AN_0$ | 102 to 95 | 112 to 105 | Input | Analog 7 to 0: Analog input pins. |
| | $\overline{ADTRG}$ | 92 | 102 | Input | A/D conversion external trigger input: Pin for input of an external trigger to start A/D conversion. |
| D/A converter | $DA_1$, $DA_0$ | 102, 101 | 112, 111 | Output | Analog output: D/A converter analog output pins. |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| A/D converter and D/A converters | $AV_{CC}$ | 93 | 103 | Input | This is the power supply pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (+5 V). |
| | $AV_{SS}$ | 103 | 113 | Input | This is the ground pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (0 V). |
| | $V_{ref}$ | 94 | 104 | Input | This is the reference voltage input pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (+5 V). |
| I/O ports | $P1_7$ to $P1_0$ | 105 to 112 | 115 to 122 | I/O | Port 1: An 8-bit I/O port. Input or output can be designated for each bit by means of the port 1 data direction register (P1DDR). |
| | $P2_7$ to $P2_0$ | 64 to 71 | 72 to 79 | I/O | Port 2: An 8-bit I/O port. Input or output can be designated for each bit by means of the port 2 data direction register (P2DDR). |
| | $P3_5$ to $P3_0$ | 58 to 53 | 64 to 59 | I/O | Port 3: A 6-bit I/O port. Input or output can be designated for each bit by means of the port 3 data direction register (P3DDR). |
| | $P4_7$ to $P4_0$ | 102 to 95 | 112 to 105 | Input | Port 4: An 8-bit input port. |
| | $P5_3$ to $P5_0$ | 92 to 89 | 102, 101, 98, 97 | I/O | Port 5: A 4-bit I/O port. Input or output can be designated for each bit by means of the port 5 data direction register (P5DDR). |

**Table 1-3 Pin Functions (cont)**

| Type | Symbol | Pin No. TFP-120 | Pin No. FP-128 | I/O | Name and Function |
|---|---|---|---|---|---|
| I/O ports | $P6_7$ to $P6_0$ | 29 to 32, 63 to 60 | 33, 34, 37, 38, 71 to 69, 66 | I/O | Port 6: An 8-bit I/O port. Input or output can be designated for each bit by means of the port 6 data direction register (P6DDR). |
| | $PA_7$ to $PA_0$ | 28 to 25, 23 to 20 | 32 to 29, 27 to 24 | I/O | Port A: An 8-bit I/O port. Input or output can be designated for each bit by means of the port A data direction register (PADDR). |
| | $PB_7$ to $PB_0$ | 19 to 16, 14 to 11 | 23 to 20, 18 to 15 | I/O | Port B: An 8-bit I/O port. Input or output can be designated for each bit by means of the port B data direction register (PBDDR). |
| | $PC_7$ to $PC_0$ | 10 to 7, 5 to 2 | 14 to 11, 9 to 6 | I/O | Port C: An 8-bit I/O port. Input or output can be designated for each bit by means of the port C data direction register (PCDDR). |
| | $PD_7$ to $PD_0$ | 51 to 48, 46 to 43 | 57 to 54, 52 to 49 | I/O | Port D: An 8-bit I/O port. Input or output can be designated for each bit by means of the port D data direction register (PDDDR). |
| | $PE_7$ to $PE_0$ | 42 to 39, 37 to 34 | 48 to 45, 43 to 40 | I/O | Port E: An 8-bit I/O port. Input or output can be designated for each bit by means of the port E data direction register (PEDDR). |
| | $PF_7$ to $PF_0$ | 80, 82 to 88 | 88, 90 to 96 | I/O | Port F: An 8-bit I/O port. Input or output can be designated for each bit by means of the port F data direction register (PFDDR). |
| | $PG_4$ to $PG_0$ | 120 to 116 | 2, 1, 128 to 126 | I/O | Port G: A 5-bit I/O port. Input or output can be designated for each bit by means of the port G data direction register (PGDDR). |

# Section 2 CPU

## 2.1 Overview

The H8S/2600 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward-compatible with the H8/300 and H8/300H CPUs. The H8S/2600 CPU has sixteen 16-bit general registers, can address a 16-Mbyte (architecturally 4-Gbyte) linear address space, and is ideal for realtime control.

### 2.1.1 Features

The H8S/2600 CPU has the following features.

- Upward-compatible with H8/300 and H8/300H CPUs
  - Can execute H8/300 and H8/300H object programs
- General-register architecture
  - Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)
- Sixty-nine basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
  - Multiply-and-accumulate instruction
- Eight addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  - Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Memory indirect [@@aa:8]
- 16-Mbyte address space
  - Program: 16 Mbytes
  - Data: 16 Mbytes (4 Gbytes architecturally)

- High-speed operation
  - All frequently-used instructions execute in one or two states
  - Maximum clock rate : 20 MHz
  - 8/16/32-bit register-register add/subtract : 50 ns
  - 8 × 8-bit register-register multiply : 150 ns
  - 16 ÷ 8-bit register-register divide : 600 ns
  - 16 × 16-bit register-register multiply : 200 ns
  - 32 ÷ 16-bit register-register divide : 1000 ns

- Two CPU operating modes
  - Normal mode
  - Advanced mode

- Power-down state
  - Transition to power-down state by SLEEP instruction
  - CPU clock speed selection

### 2.1.2 Differences from H8/300 CPU

In comparison to the H8/300 CPU, the H8S/2600 CPU has the following enhancements.

- More general registers and control registers
  - Eight 16-bit expanded registers, and one 8-bit and two 32-bit control registers, have been added.

- Expanded address space
  - Normal mode supports the same 64-kbyte address space as the H8/300 CPU.
  - Advanced mode supports a maximum 16-Mbyte address space.

- Enhanced addressing
  - The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.

- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Signed multiply and divide instructions have been added.
  - A multiply-and-accumulate instruction has been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

### 2.1.3 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2600 CPU has the following enhancements.

- Additional control register
  - One 8-bit and two 32-bit control registers have been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - A multiply-and-accumulate instruction has been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

## 2.2 CPU Operating Modes

The H8S/2600 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte total address space (architecturally a maximum 16-Mbyte program area and a maximum of 4 Gbytes for program and data areas combined). The mode is selected by the mode pins of the microcontroller.

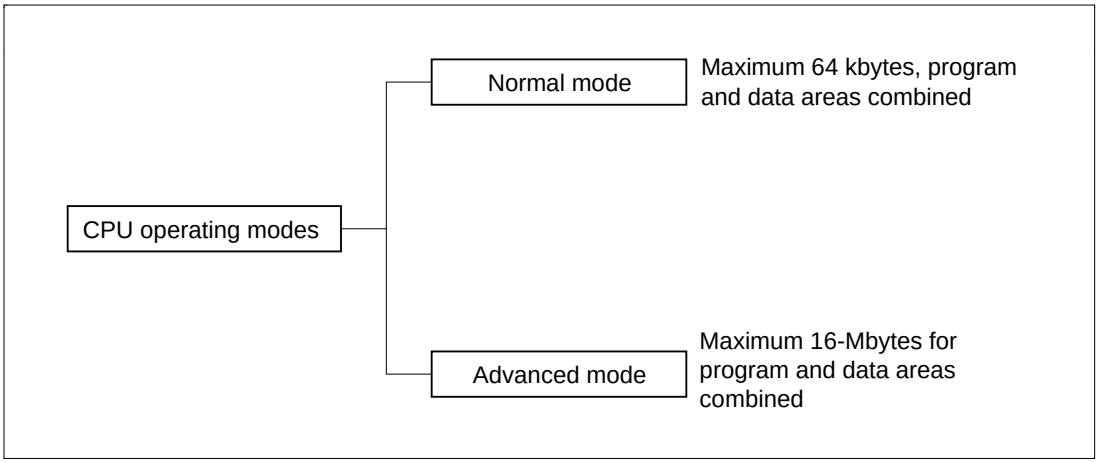



**Figure 2-1 CPU Operating Modes**

### (1) Normal Mode

The exception vector table and stack have the same structure as in the H8/300 CPU.

**Address Space:** A maximum address space of 64 kbytes can be accessed.

**Extended Registers (En):** The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When En is used as a 16-bit register it can contain any value, even when the corresponding general register (Rn) is used as an address register. If the general register is referenced in the register indirect addressing mode with pre-decrement (@–Rn) or post-increment (@Rn+) and a carry or borrow occurs, however, the value in the corresponding extended register (En) will be affected.

**Instruction Set:** All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

**Exception Vector Table and Memory Indirect Branch Addresses:** In normal mode the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits (figure 2-2). The exception vector table differs depending on the microcontroller. For details of the exception vector table, see section 4, Exception Handling.

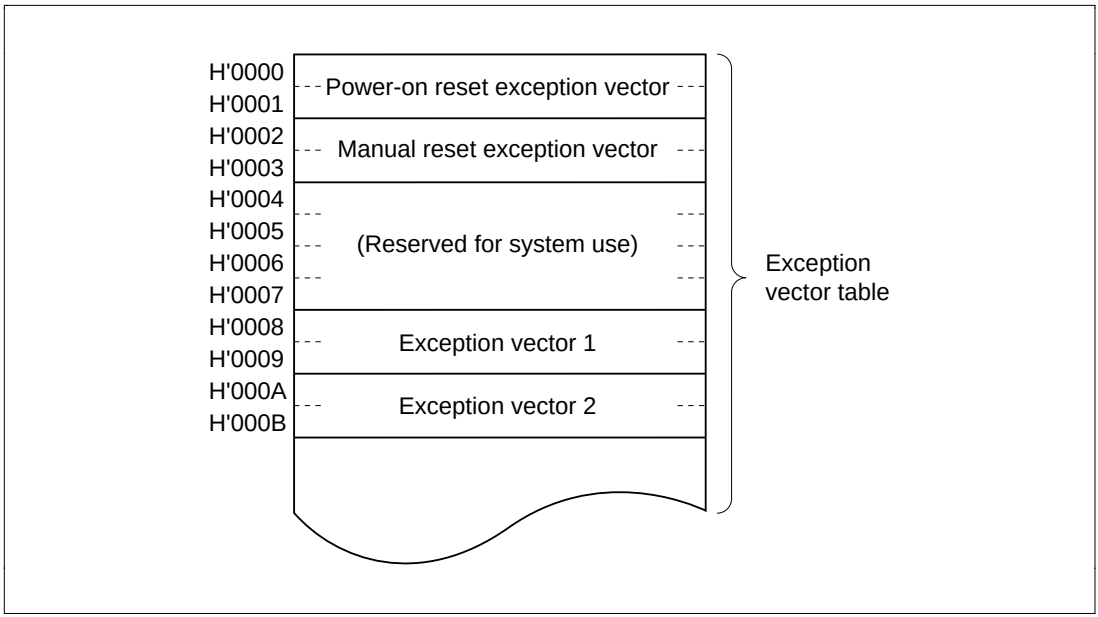
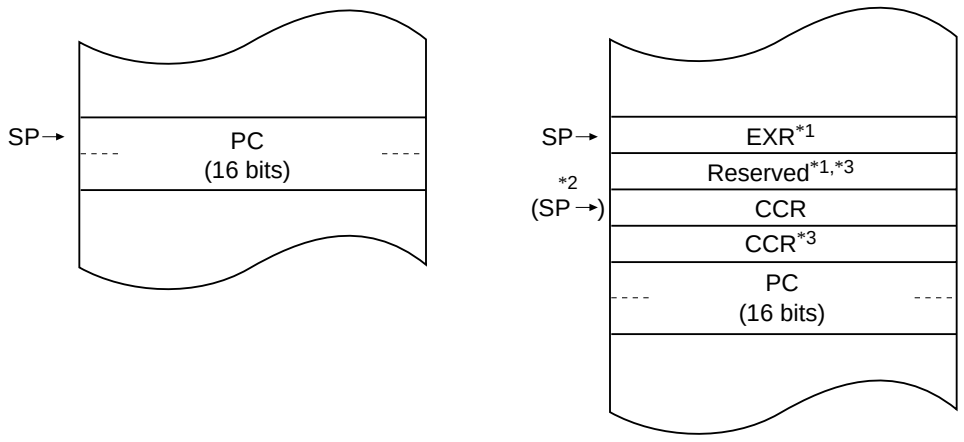



**Figure 2-2 Exception Vector Table (Normal Mode)**

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In normal mode the operand is a 16-bit word operand, providing a 16-bit branch address. Branch addresses can be stored in the top area from H'0000 to H'00FF. Note that this area is also used for the exception vector table.

**Stack Structure:** When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2-3. When EXR is invalid, it is not pushed onto the stack. For details, see section 4, Exception Handling.




**(a) Subroutine Branch** **(b) Exception Handling**

Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

**Figure 2-3 Stack Structure in Normal Mode**

### (2) Advanced Mode

**Address Space:** Linear access is provided to a 16-Mbyte maximum address space (architecturally a maximum 16-Mbyte program area and a maximum 4-Gbyte data area, with a maximum of 4 Gbytes for program and data areas combined).

**Extended Registers (En):** The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.

**Instruction Set:** All instructions and addressing modes can be used.

**Exception Vector Table and Memory Indirect Branch Addresses:** In advanced mode the top area starting at H'00000000 is allocated to the exception vector table in units of 32 bits. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (figure 2-4). For details of the exception vector table, see section 4, Exception Handling.

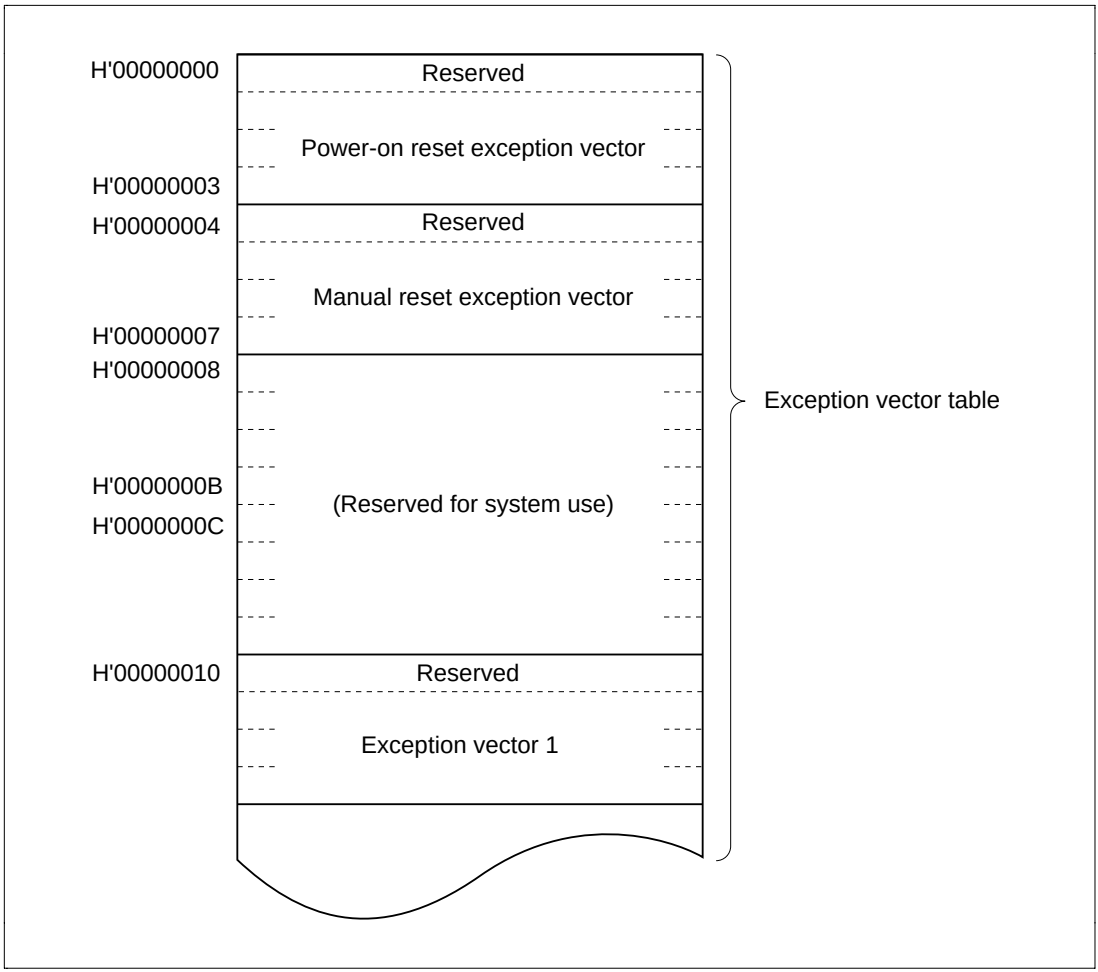
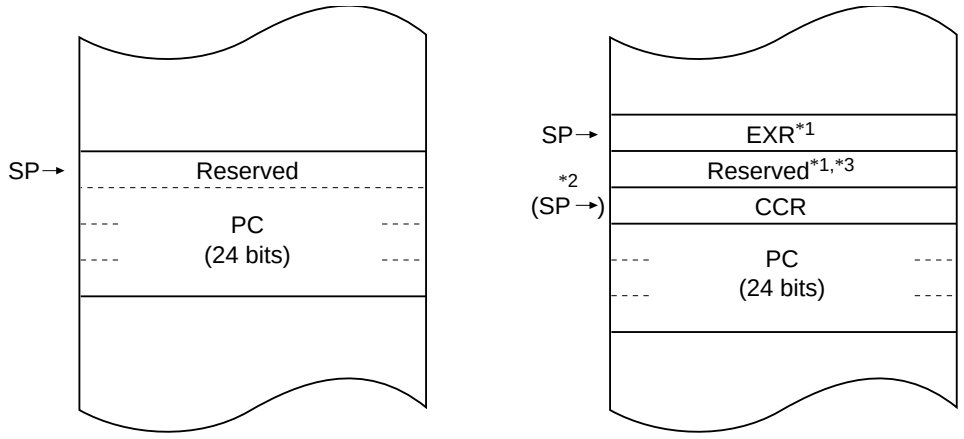



**Figure 2-4 Exception Vector Table (Advanced Mode)**

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits are a reserved area that is regarded as H'00. Branch addresses can be stored in the area from H'00000000 to H'000000FF. Note that the first part of this range is also the exception vector table.

**Stack Structure:** In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2-5. When EXR is invalid, it is not pushed onto the stack. For details, see section 4, Exception Handling.




**(a) Subroutine Branch** **(b) Exception Handling**

Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

**Figure 2-5 Stack Structure in Advanced Mode**

## 2.3 Address Space

Figure 2-6 shows a memory map of the H8S/2600 CPU. The H8S/2600 CPU provides linear access to a maximum 64-kbyte address space in normal mode, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode.

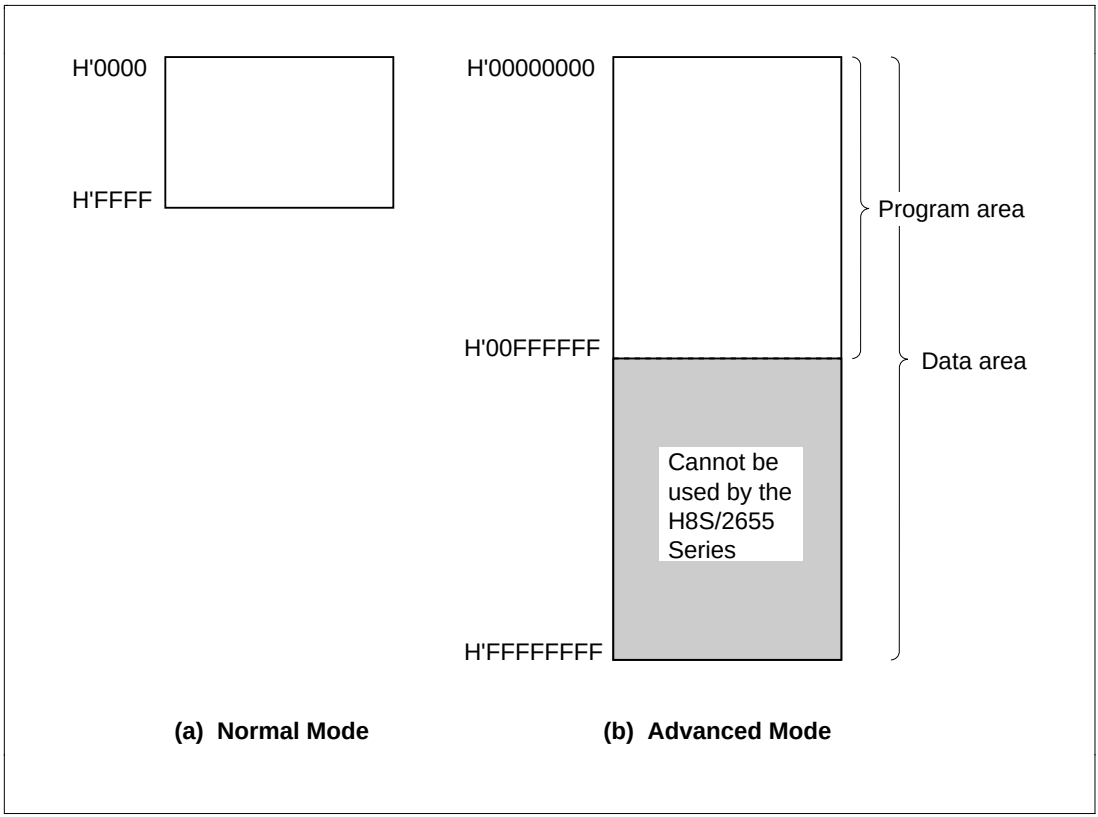



**Figure 2-6 Memory Map**

## 2.4 Register Configuration

### 2.4.1 Overview

The CPU has the internal registers shown in figure 2-7. There are two types of registers: general registers and control registers.

**General Registers (Rn) and Extended Registers (En)**

| | 15 07 | 07 0 | |
|---|---|---|---|
| ER0 | E0 | R0H | R0L |
| ER1 | E1 | R1H | R1L |
| ER2 | E2 | R2H | R2L |
| ER3 | E3 | R3H | R3L |
| ER4 | E4 | R4H | R4L |
| ER5 | E5 | R5H | R5L |
| ER6 | E6 | R6H | R6L |
| ER7 (SP) | E7 | R7H | R7L |

**Control Registers (CR)**

**Legend**

| | | | |
|---|---|---|---|
| SP: | Stack pointer | H: | Half-carry flag |
| PC: | Program counter | U: | User bit |
| EXR: | Extended control register | N: | Negative flag |
| T: | Trace bit | Z: | Zero flag |
| I2 to I0: | Interrupt mask bits | V: | Overflow flag |
| CCR: | Condition-code register | C: | Carry flag |
| I: | Interrupt mask bit | MAC: | Multiply-accumulate register |
| UI: | User bit or interrupt mask bit | | |

**Figure 2-7 CPU Registers**

### 2.4.2 General Registers

The CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

The ER registers divide into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

Figure 2-8 illustrates the usage of the general registers. The usage of each register can be selected independently.

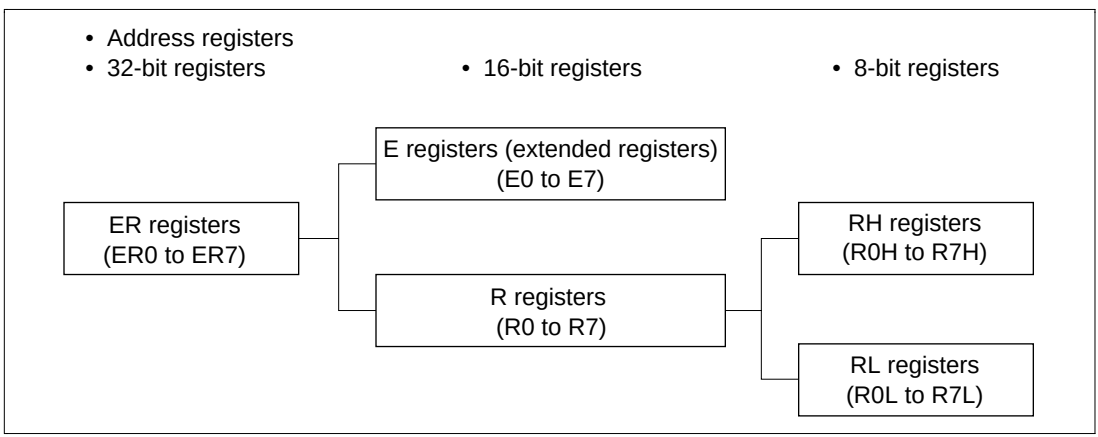



**Figure 2-8 Usage of General Registers**

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2-9 shows the stack.

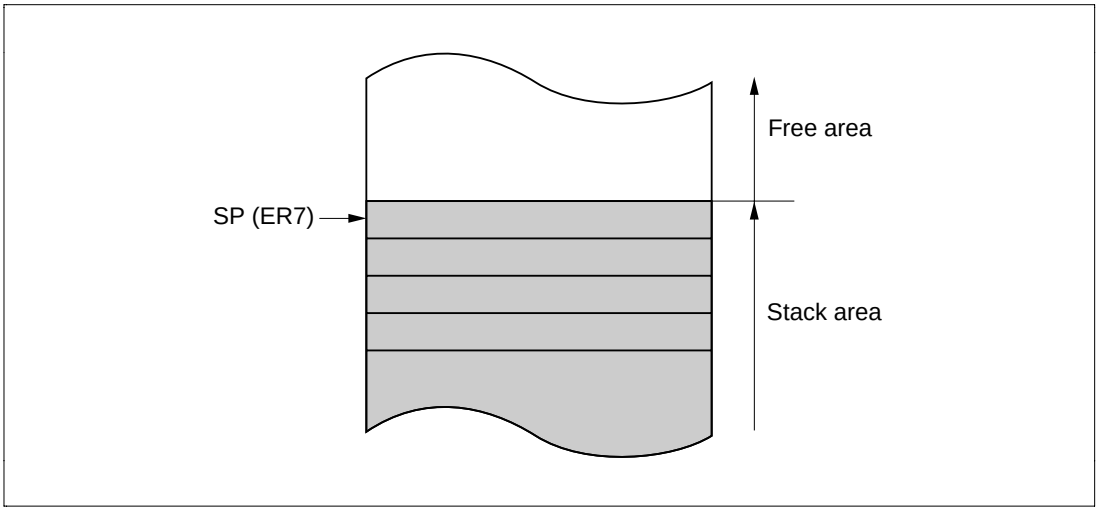



**Figure 2-9 Stack**

### 2.4.3 Control Registers

The control registers are the 24-bit program counter (PC), 8-bit extended control register (EXR), 8-bit condition-code register (CCR), and 64-bit multiply-accumulate register (MAC).

**(1) Program Counter (PC):** This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched, the least significant PC bit is regarded as 0.)

**(2) Extended Control Register (EXR):** This 8-bit register contains the trace bit (T) and three interrupt mask bits (I2 to I0).

**Bit 7—Trace Bit (T):** Selects trace mode. When this bit is cleared to 0, instructions are executed in sequence. When this bit is set to 1, a trace exception is generated each time an instruction is executed.

**Bits 6 to 3—Reserved:** These bits are reserved. They are always read as 1.

**Bits 2 to 0—Interrupt Mask Bits (I2 to I0):** These bits designate the interrupt mask level (0 to 7). For details, refer to section 5, Interrupt Controller.

Operations can be performed on the EXR bits by the LDC, STC, ANDC, ORC, and XORC instructions. All interrupts, including NMI, are disabled for three states after one of these instructions is executed, except for STC.

**(3) Condition-Code Register (CCR):** This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

**Bit 7—Interrupt Mask Bit (I):** Masks interrupts other than NMI when set to 1. (NMI is accepted regardless of the I bit setting.) The I bit is set to 1 by hardware at the start of an exception-handling sequence. For details, refer to section 5, Interrupt Controller.

**Bit 6—User Bit or Interrupt Mask Bit (UI):** Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit can also be used as an interrupt mask bit. For details, refer to section 5, Interrupt Controller.

**Bit 5—Half-Carry Flag (H):** When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.

**Bit 4—User Bit (U):** Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.

**Bit 3—Negative Flag (N):** Stores the value of the most significant bit (sign bit) of data.

**Bit 2—Zero Flag (Z):** Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.

**Bit 1—Overflow Flag (V):** Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.

**Bit 0—Carry Flag (C):** Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:

- Add instructions, to indicate a carry
- Subtract instructions, to indicate a borrow
- Shift and rotate instructions, to store the value shifted out of the end bit

The carry flag is also used as a bit accumulator by bit manipulation instructions.

Some instructions leave some or all of the flag bits unchanged. For the action of each instruction on the flag bits, refer to Appendix A.1, List of Instructions.

Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branching conditions for conditional branch (Bcc) instructions.

**(4) Multiply-Accumulate Register (MAC):** This 64-bit register stores the results of multiply-and-accumulate operations. It consists of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid; the upper bits are a sign extension.

### 2.4.4 Initial Register Values

Reset exception handling loads the CPU's program counter (PC) from the vector table, clears the trace bit in EXR to 0, and sets the interrupt mask bits in CCR and EXR to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (ER7) is not initialized. The stack pointer should therefore be initialized by an MOV.L instruction executed immediately after a reset.

## 2.5 Data Formats

The CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

### 2.5.1 General Register Data Formats

Figure 2-10 shows the data formats in general registers.

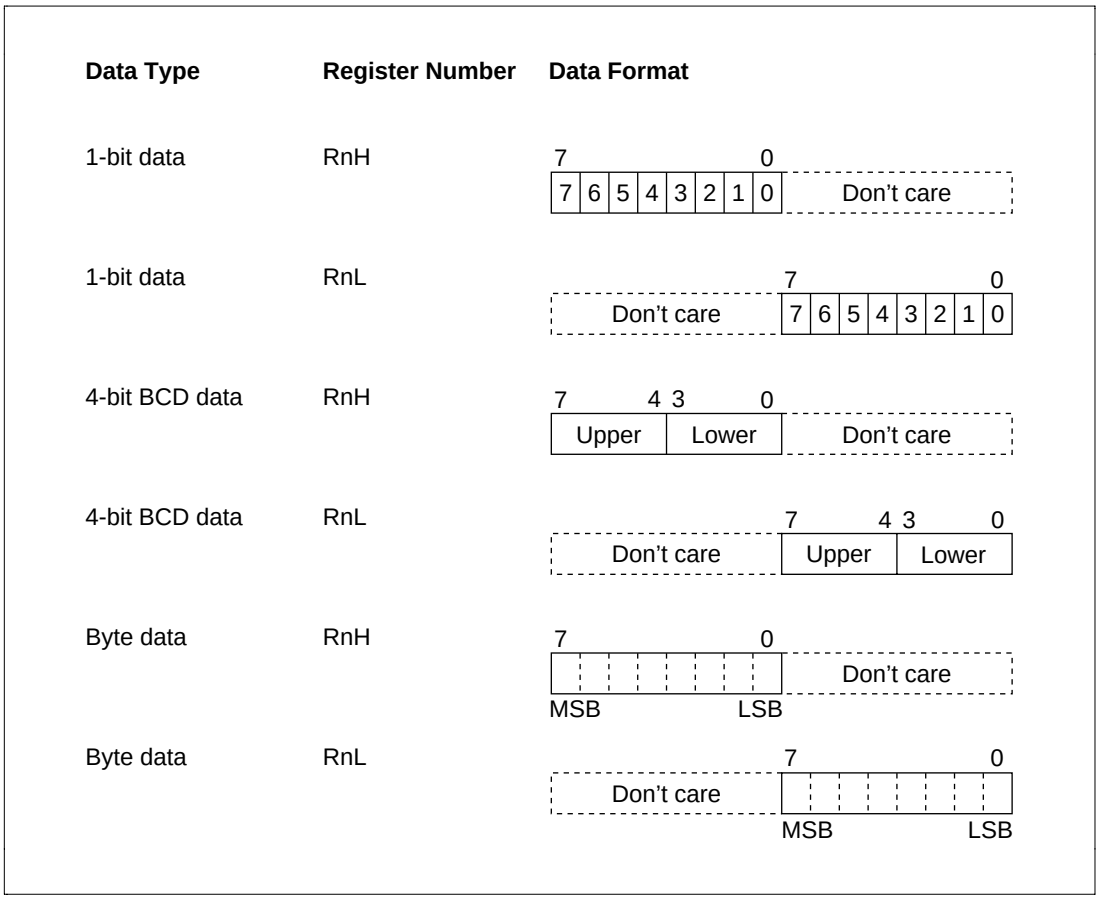



**Figure 2-10 General Register Data Formats**

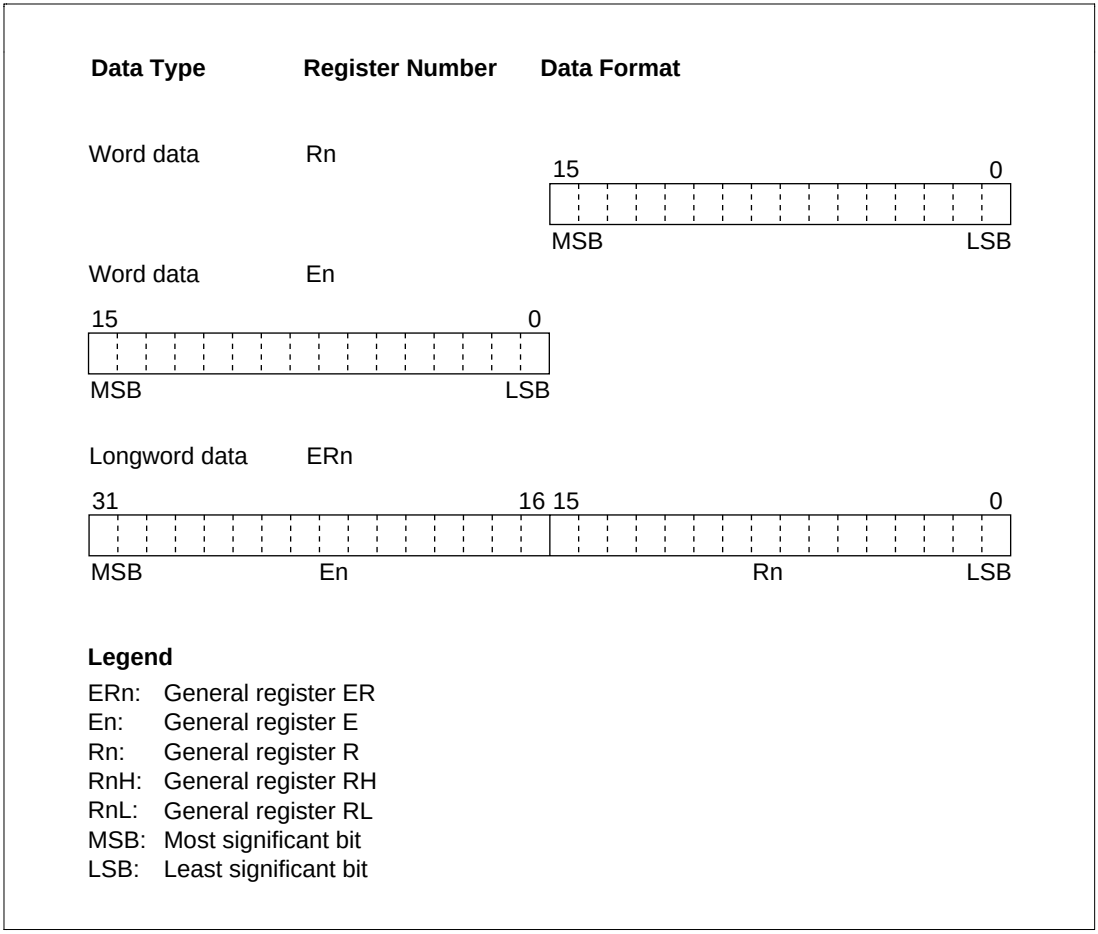



**Figure 2-10 General Register Data Formats (cont)**

### 2.5.2 Memory Data Formats

Figure 2-11 shows the data formats in memory. The CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

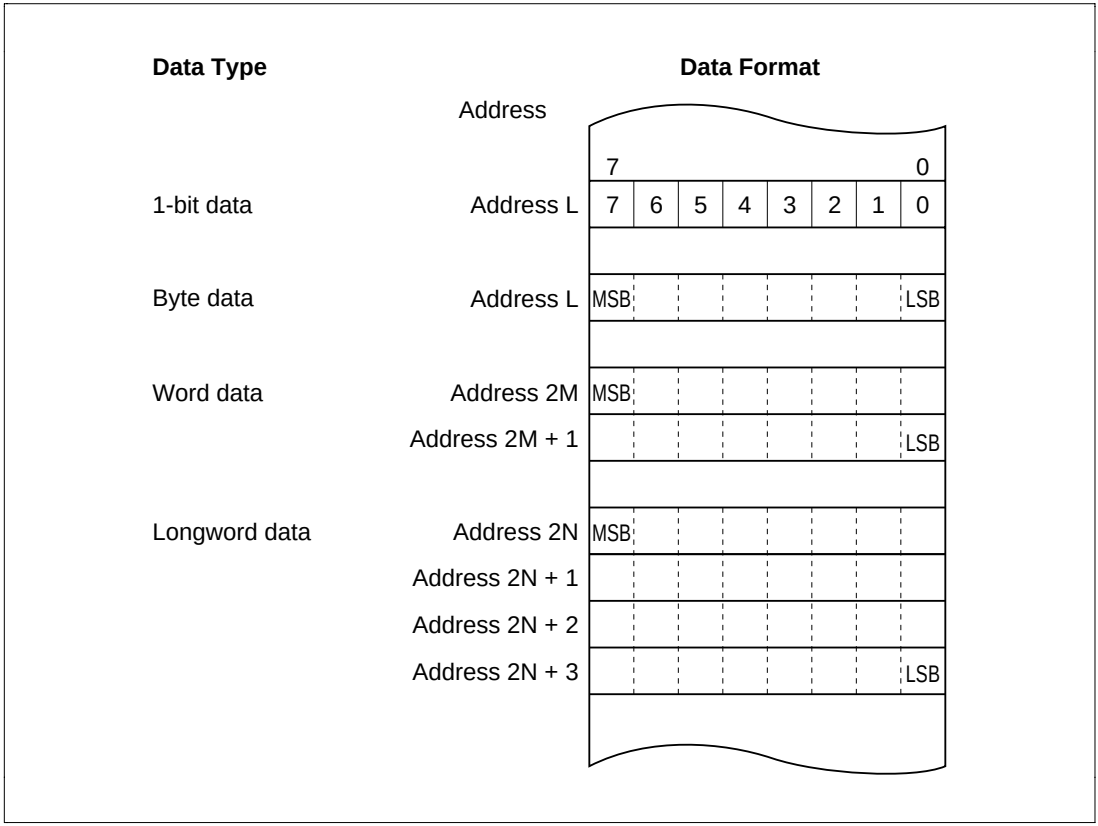



**Figure 2-11 Memory Data Formats**

When ER7 is used as an address register to access the stack, the operand size should be word size or longword size.

## 2.6 Instruction Set

### 2.6.1 Overview

The H8S/2600 CPU has 69 types of instructions. The instructions are classified by function in table 2-1.

**Table 2-1 Instruction Classification**

| Function | Instructions | Size | Types |
|---|---|---|---|
| Data transfer | MOV | BWL | 5 |
| | POP*[1], PUSH*[1] | WL | |
| | LDM, STM | L | |
| | MOVFPE, MOVTPE | B | |
| Arithmetic operations | ADD, SUB, CMP, NEG | BWL | 23 |
| | ADDX, SUBX, DAA, DAS | B | |
| | INC, DEC | BWL | |
| | ADDS, SUBS | L | |
| | MULXU, DIVXU, MULXS, DIVXS | BW | |
| | EXTU, EXTS | WL | |
| | TAS | B | |
| | MAC, LDMAC, STMAC, CLRMAC | — | |
| Logic operations | AND, OR, XOR, NOT | BWL | 4 |
| Shift | SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR | BWL | 8 |
| Bit manipulation | BSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR | B | 14 |
| Branch | Bcc*[2], JMP, BSR, JSR, RTS | — | 5 |
| System control | TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP | — | 9 |
| Block data transfer | EEPMOV | — | 1 |

Notes: B-byte size; W-word size; L-longword size.

1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP. POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Bcc is the general name for conditional branch instructions.

### 2.6.2 Instructions and Addressing Modes

Table 2-2 indicates the combinations of instructions and addressing modes that the H8S/2600 CPU can use.

**Table 2.2 Combinations of Instructions and Addressing Modes**

| Function | Instruction | Addressing Modes | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d:16,ERn) | @(d:32,ERn) | @–ERn/@ERn+ | @aa:8 | @aa:16 | @aa:24 | @aa:32 | @(d:8,PC) | @(d:16,PC) | @@aa:8 | — |
| **Data transfer** | MOV | BWL | BWL | BWL | BWL | BWL | BWL | B | BWL | — | BWL | — | — | — | — |
| | POP, PUSH | — | — | — | — | — | — | — | — | — | — | — | — | — | WL |
| | LDM, STM | — | — | — | — | — | — | — | — | — | — | — | — | — | L |
| | MOVEPE, MOVTPE | — | — | — | — | — | — | — | B | — | — | — | — | — | — |
| **Arithmetic operations** | ADD, CMP | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | SUB | WL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDX, SUBX | B | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDS, SUBS | — | L | — | — | — | — | — | — | — | — | — | — | — | — |
| | INC, DEC | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | DAA, DAS | — | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXU, DIVXU | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXS, DIVXS | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | NEG | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | EXTU, EXTS | — | WL | — | — | — | — | — | — | — | — | — | — | — | — |
| | TAS | — | — | B | — | — | — | — | — | — | — | — | — | — | — |
| | MAC | — | — | — | — | — | ○ | — | — | — | — | — | — | — | — |
| | CLRMAC | — | — | — | — | — | — | — | — | — | — | — | — | ○ | — |
| | LDMAC, STMAC | — | L | — | — | — | — | — | — | — | — | — | — | — | — |

## Table 2.2 Combinations of Instructions and Addressing Modes (cont)

| Function | Instruction | Addressing Modes | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d:16,ERn) | @(d:32,ERn) | @–ERn/@ERn+ | @aa:8 | @aa:16 | @aa:24 | @aa:32 | @(d:8,PC) | @(d:16,PC) | @@aa:8 | — |
| Logic operations | AND, OR, XOR | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | NOT | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Shift | | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Bit manipulation | | — | B | B | — | — | — | B | B | — | B | — | — | — | — |
| Branch | Bcc, BSR | — | — | — | — | — | — | — | — | — | — | ○ | ○ | — | — |
| | JMP, JSR | — | — | — | — | — | — | — | — | ○ | — | — | — | ○ | — |
| | RTS | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| System control | TRAPA | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | RTE | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | SLEEP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | LDC | B | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | STC | — | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | ANDC, ORC, XORC | B | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | NOP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| Block data transfer | | — | — | — | — | — | — | — | — | — | — | — | — | — | BW |

**Legend**
B: Byte
W: Word
L: Longword

### 2.6.3 Table of Instructions Classified by Function

Table 2-3 summarizes the instructions in each functional category. The notation used in table 2-3 is defined below.

**Operation Notation**

| | |
|---|---|
| Rd | General register (destination)* |
| Rs | General register (source)* |
| Rn | General register* |
| ERn | General register (32-bit register) |
| MAC | Multiply-accumulate register (32-bit register) |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extended control register |
| CCR | Condition-code register |
| N | N (negative) flag in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| disp | Displacement |
| + | Addition |
| – | Subtraction |
| × | Multiplication |
| ÷ | Division |
| ∧ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Logical exclusive OR |
| → | Move |
| ¬ | NOT (logical complement) |
| :8/:16/:24/:32 | 8-, 16-, 24-, or 32-bit length |

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

**Table 2-3 Instructions Classified by Function**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Data transfer | MOV | B/W/L | (EAs) → Rd, Rs → (Ead)<br>Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. |
| | MOVFPE | B | Cannot be used in the H8S/2655 Series. |
| | MOVTPE | B | Cannot be used in the H8S/2655 Series. |
| | POP | W/L | @SP+ → Rn<br>Pops a register from the stack. POP.W Rn is identical to MOV.W @SP+, Rn. POP.L ERn is identical to MOV.L @SP+, ERn. |
| | PUSH | W/L | Rn → @–SP<br>Pushes a register onto the stack. PUSH.W Rn is identical to MOV.W Rn, @–SP. PUSH.L ERn is identical to MOV.L ERn, @–SP. |
| | LDM | L | @SP+ → Rn (register list)<br>Pops two or more general registers from the stack. |
| | STM | L | Rn (register list) → @–SP<br>Pushes two or more general registers onto the stack. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Arithmetic operations | ADD<br>SUB | B/W/L | Rd ± Rs → Rd, Rd ± #IMM → Rd<br>Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register. (Immediate byte data cannot be subtracted from byte data in a general register. Use the SUBX or ADD instruction.) |
| | ADDX<br>SUBX | B | Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd<br>Performs addition or subtraction with carry or borrow on byte data in two general registers, or on immediate data and data in a general register. |
| | INC<br>DEC | B/W/L | Rd ± 1 → Rd, Rd ± 2 → Rd<br>Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.) |
| | ADDS<br>SUBS | L | Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd<br>Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register. |
| | DAA<br>DAS | B | Rd decimal adjust → Rd<br>Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 4-bit BCD data. |
| | MULXU | B/W | Rd × Rs → Rd<br>Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits. |
| | MULXS | B/W | Rd × Rs → Rd<br>Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits. |
| | DIVXU | B/W | Rd ÷ Rs → Rd<br>Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Arithmetic operations | DIVXS | B/W | Rd ÷ Rs → Rd<br>Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder. |
| | CMP | B/W/L | Rd – Rs, Rd – #IMM<br>Compares data in a general register with data in another general register or with immediate data, and sets CCR bits according to the result. |
| | NEG | B/W/L | 0 – Rd → Rd<br>Takes the two's complement (arithmetic complement) of data in a general register. |
| | EXTU | W/L | Rd (zero extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left. |
| | EXTS | W/L | Rd (sign extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit. |
| | TAS | B | @ERd – 0, 1 → (<bit 7> of @Erd)<br>Tests memory contents, and sets the most significant bit (bit 7) to 1. |
| | MAC | — | (EAs) × (EAd) + MAC → MAC<br>Performs signed multiplication on memory contents and adds the result to the multiply-accumulate register. The following operations can be performed:<br>16 bits × 16 bits + 32 bits → 32 bits, saturating<br>16 bits × 16 bits + 42 bits → 42 bits, non-saturating |
| | CLRMAC | — | 0 → MAC<br>Clears the multiply-accumulate register to zero. |
| | LDMAC<br>STMAC | L | Rs → MAC, MAC → Rd<br>Transfers data between a general register and a multiply-accumulate register. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Logic operations | AND | B/W/L | Rd ∧ Rs → Rd, Rd ∧ #IMM → Rd<br>Performs a logical AND operation on a general register and another general register or immediate data. |
| | OR | B/W/L | Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd<br>Performs a logical OR operation on a general register and another general register or immediate data. |
| | XOR | B/W/L | Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd<br>Performs a logical exclusive OR operation on a general register and another general register or immediate data. |
| | NOT | B/W/L | ¬ (Rd) → (Rd)<br>Takes the one's complement of general register contents. |
| Shift operations | SHAL<br>SHAR | B/W/L | Rd (shift) → Rd<br>Performs an arithmetic shift on general register contents.<br>1-bit or 2-bit shift is possible. |
| | SHLL<br>SHLR | B/W/L | Rd (shift) → Rd<br>Performs a logical shift on general register contents.<br>1-bit or 2-bit shift is possible. |
| | ROTL<br>ROTR | B/W/L | Rd (rotate) → Rd<br>Rotates general register contents.<br>1-bit or 2-bit rotation is possible. |
| | ROTXL<br>ROTXR | B/W/L | Rd (rotate) → Rd<br>Rotates general register contents through the carry flag.<br>1-bit or 2-bit rotation is possible. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Bit-manipulation instructions | BSET | B | 1 → (<bit-No.> of <EAd>)<br>Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BCLR | B | 0 → (<bit-No.> of <EAd>)<br>Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BNOT | B | ¬ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>)<br>Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BTST | B | ¬ (<bit-No.> of <EAd>) → Z<br>Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BAND | B | C ∧ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIAND | B | C ∧ ¬ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BOR | B | C ∨ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIOR | B | C ∨ ¬ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |

Note: * Size refers to the operand size.
B: Byte

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Bit-manipulation instructions | BXOR | B | C ⊕ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIXOR | B | C ⊕ ¬ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BLD | B | (<bit-No.> of <EAd>) → C<br>Transfers a specified bit in a general register or memory operand to the carry flag. |
| | BILD | B | ¬ (<bit-No.> of <EAd>) → C<br>Transfers the inverse of a specified bit in a general register or memory operand to the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BST | B | C → (<bit-No.> of <EAd>)<br>Transfers the carry flag value to a specified bit in a general register or memory operand. |
| | BIST | B | ¬ C → (<bit-No.> of <EAd>)<br>Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand.<br>The bit number is specified by 3-bit immediate data. |

Note: * Size refers to the operand size.
B: Byte

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Branch instructions | Bcc | — | Branches to a specified address if a specified condition is true. The branching conditions are listed below. |

| Mnemonic | Description | Condition |
|---|---|---|
| BRA(BT) | Always (true) | Always |
| BRN(BF) | Never (false) | Never |
| BHI | High | $C \vee Z = 0$ |
| BLS | Low or same | $C \vee Z = 1$ |
| BCC(BHS) | Carry clear (high or same) | $C = 0$ |
| BCS(BLO) | Carry set (low) | $C = 1$ |
| BNE | Not equal | $Z = 0$ |
| BEQ | Equal | $Z = 1$ |
| BVC | Overflow clear | $V = 0$ |
| BVS | Overflow set | $V = 1$ |
| BPL | Plus | $N = 0$ |
| BMI | Minus | $N = 1$ |
| BGE | Greater or equal | $N \oplus V = 0$ |
| BLT | Less than | $N \oplus V = 1$ |
| BGT | Greater than | $Z \vee (N \oplus V) = 0$ |
| BLE | Less or equal | $Z \vee (N \oplus V) = 1$ |

| | | | |
|---|---|---|---|
| | JMP | — | Branches unconditionally to a specified address. |
| | BSR | — | Branches to a subroutine at a specified address. |
| | JSR | — | Branches to a subroutine at a specified address. |
| | RTS | — | Returns from a subroutine |

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| System control instructions | TRAPA | — | Starts trap-instruction exception handling. |
| | RTE | — | Returns from an exception-handling routine. |
| | SLEEP | — | Causes a transition to a power-down state. |
| | LDC | B/W | (EAs) → CCR, (EAs) → EXR<br>Moves the source operand contents or immediate data to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| | STC | B/W | CCR → (EAd), EXR → (EAd)<br>Transfers CCR or EXR contents to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| | ANDC | B | CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR<br>Logically ANDs the CCR or EXR contents with immediate data. |
| | ORC | B | CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR<br>Logically ORs the CCR or EXR contents with immediate data. |
| | XORC | B | CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR<br>Logically exclusive-ORs the CCR or EXR contents with immediate data. |
| | NOP | — | PC + 2 → PC<br>Only increments the program counter. |

Note: * Size refers to the operand size.
B: Byte
W: Word

**Table 2-3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Block data transfer instruction | EEPMOV.B | — | if R4L ≠ 0 then<br>Repeat @ER5+ → @ER6+<br>R4L–1 → R4L<br>Until R4L = 0<br>else next; |
| | EEPMOV.W | — | if R4 ≠ 0 then<br>Repeat @ER5+ → @ER6+<br>R4–1 → R4<br>Until R4 = 0<br>else next;<br>Transfers a data block according to parameters set in general registers R4L or R4, ER5, and ER6.<br>R4L or R4: size of block (bytes)<br>ER5: starting source address<br>ER6: starting destination address<br>Execution of the next instruction begins as soon as the transfer is completed. |

### 2.6.4 Basic Instruction Formats

The H8S/2655 Series instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

**(1) Operation Field:** Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.

**(2) Register Field:** Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.

**(3) Effective Address Extension:** Eight, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.

**(4) Condition Field:** Specifies the branching condition of Bcc instructions.

Figure 2-12 shows examples of instruction formats.

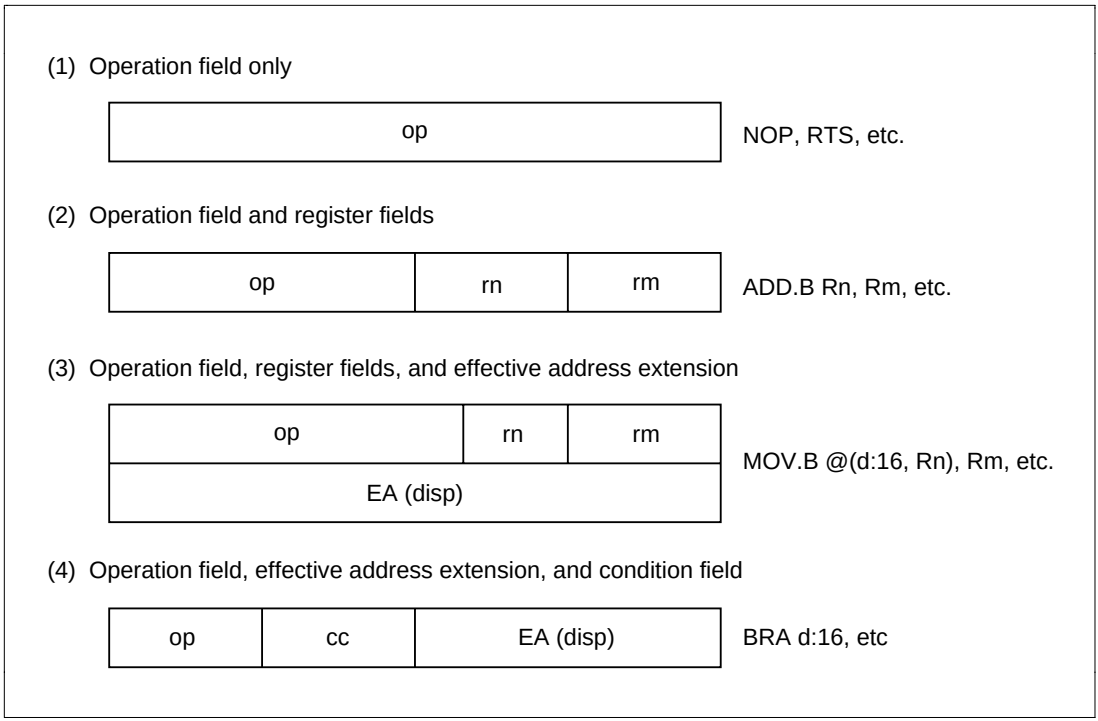



**Figure 2-12 Instruction Formats (Examples)**

## 2.7 Addressing Modes and Effective Address Calculation

### 2.7.1 Addressing Mode

The CPU supports the eight addressing modes listed in table 2-4. Each instruction uses a subset of these addressing modes. Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

**Table 2-4 Addressing Modes**

| No. | Addressing Mode | Symbol |
|---|---|---|
| 1 | Register direct | Rn |
| 2 | Register indirect | @ERn |
| 3 | Register indirect with displacement | @(d:16,ERn)/@(d:32,ERn) |
| 4 | Register indirect with post-increment<br>Register indirect with pre-decrement | @ERn+<br>@–ERn |
| 5 | Absolute address | @aa:8/@aa:16/@aa:24/@aa:32 |
| 6 | Immediate | #xx:8/#xx:16/#xx:32 |
| 7 | Program-counter relative | @(d:8,PC)/@(d:16,PC) |
| 8 | Memory indirect | @@aa:8 |

**(1) Register Direct—Rn:** The register field of the instruction specifies an 8-, 16-, or 32-bit general register containing the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

**(2) Register Indirect—@ERn:** The register field of the instruction code specifies an address register (ERn) which contains the address of the operand on memory. If the address is a program instruction address, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

**(3) Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn):** A 16-bit or 32-bit displacement contained in the instruction is added to an address register (ERn) specified by the register field of the instruction, and the sum gives the address of a memory operand. A 16-bit displacement is sign-extended when added.

**(4) Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn:**

- Register indirect with post-increment—@ERn+

  The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.

- Register indirect with pre-decrement—@-ERn

  The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.

**(5) Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32:** The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32).

To access data, the absolute address should be 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) long. For an 8-bit absolute address, the upper 24 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 16 bits are a sign extension. A 32-bit absolute address can access the entire address space.

A 24-bit absolute address (@aa:24) indicates the address of a program instruction. The upper 8 bits are all assumed to be 0 (H'00).

Table 2-5 indicates the accessible absolute address ranges.

**Table 2-5 Absolute Address Access Ranges**

| Absolute Address | | Normal Mode | Advanced Mode |
|---|---|---|---|
| Data address | 8 bits (@aa:8) | H'FF00 to H'FFFF | H'FFFF00 to H'FFFFFF |
| | 16 bits (@aa:16) | H'0000 to H'FFFF | H'000000 to H'007FFF,<br>H'FF8000 to H'FFFFFF |
| | 32 bits (@aa:32) | | H'000000 to H'FFFFFF |
| Program instruction address | 24 bits (@aa:24) | | |

**(6) Immediate—#xx:8, #xx:16, or #xx:32:** The instruction contains 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data as an operand.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, specifying a bit number. The TRAPA instruction contains 2-bit immediate data in its instruction code, specifying a vector address.

**(7) Program-Counter Relative—@(d:8, PC) or @(d:16, PC):** This mode is used in the Bcc and BSR instructions. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a branch address. Only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00). The PC value to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is –126 to +128 bytes (–63 to +64 words) or –32766 to +32768 bytes (–16383 to +16384 words) from the branch instruction. The resulting value should be an even number.

**(8) Memory Indirect—@@aa:8:** This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address. The upper bits of the absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in advanced mode). In normal mode the memory operand is a word operand and the branch address is 16 bits long. In advanced mode the memory operand is a longword operand, the first byte of which is assumed to be all 0 (H'00).

Note that the first part of the address range is also the exception vector area. For further details, refer to section 4, Exception Handling.

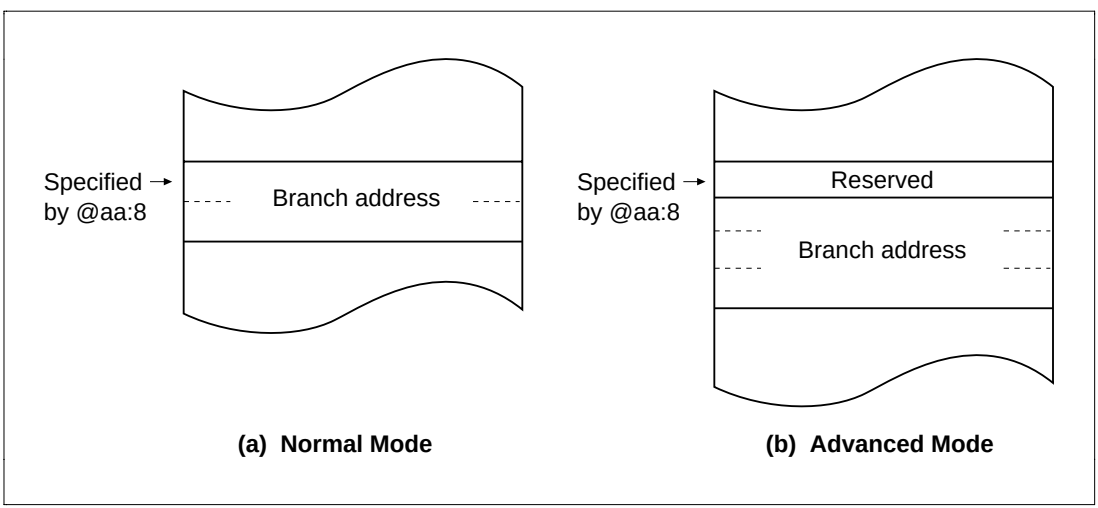



**Figure 2-13 Branch Address Specification in Memory Indirect Mode**

If an odd address is specified in word or longword memory access, or as a branch address, the least significant bit is regarded as 0, causing data to be accessed or instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)

### 2.7.2 Effective Address Calculation

Table 2-6 indicates how effective addresses are calculated in each addressing mode. In normal mode the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

**Table 2.6 Effective Address Calculation**

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 1 | Register direct (Rn)<br>op \| rm \| rm | | Operand is general register contents. |
| 2 | Register indirect (@ERn)<br>op \| r | 31 General register contents 0 | 31 Don't care 24 23 0 |
| 3 | Register indirect with displacement @(d:16, ERn) or @(d:32, ERn)<br>op \| r \| disp | 31 General register contents 0<br>31 Sign extension \| disp 0 | 31 Don't care 24 23 0 |
| 4 | Register indirect with post-increment or pre-decrement<br>• Register indirect with post-increment @ERn+<br>op \| r<br>• Register indirect with pre-decrement @–ERn<br>op \| r | 31 General register contents 0<br>1, 2, or 4<br>31 General register contents 0<br>1, 2, or 4<br>Operand Size \| Value added<br>Byte \| 1<br>Word \| 2<br>Longword \| 4 | 31 Don't care 24 23 0<br>31 Don't care 24 23 0 |

**Table 2.6 Effective Address Calculation (cont)**

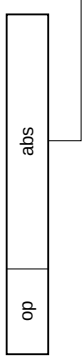
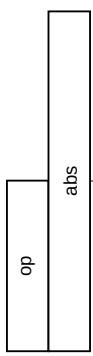
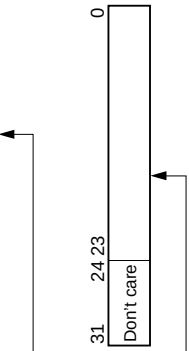
| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 5 | Absolute address<br>@aa:8<br> <br>@aa:16<br> <br>@aa:24<br>  <br>@aa:32<br>  | |  |
| 6 | Immediate #xx:8/#xx:16/#xx:32<br> | | Operand is immediate data. |

**Table 2.6 Effective Address Calculation (cont)**

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 7 | Program-counter relative<br>@(d:8, PC)/@(d:16, PC)<br>op \| disp | 23 0 PC contents<br>23 0 Sign extension \| disp | 31 24 23 0 Don't care |
| 8 | Memory indirect @@aa:8<br>• Normal mode<br>op \| abs | 31 8 7 0 H'000000 \| abs<br>15 0 Memory contents | 31 24 23 16 15 0 Don't care \| H'00 |
| | • Advanced mode<br>op \| abs | 31 8 7 0 H'000000 \| abs<br>31 0 Memory contents | 31 24 23 0 Don't care |

## 2.8 Processing States

### 2.8.1 Overview

The CPU has five main processing states: the reset state, exception handling state, program execution state, bus-released state, and power-down state. Figure 2-14 shows a diagram of the processing states. Figure 2-15 indicates the state transitions.

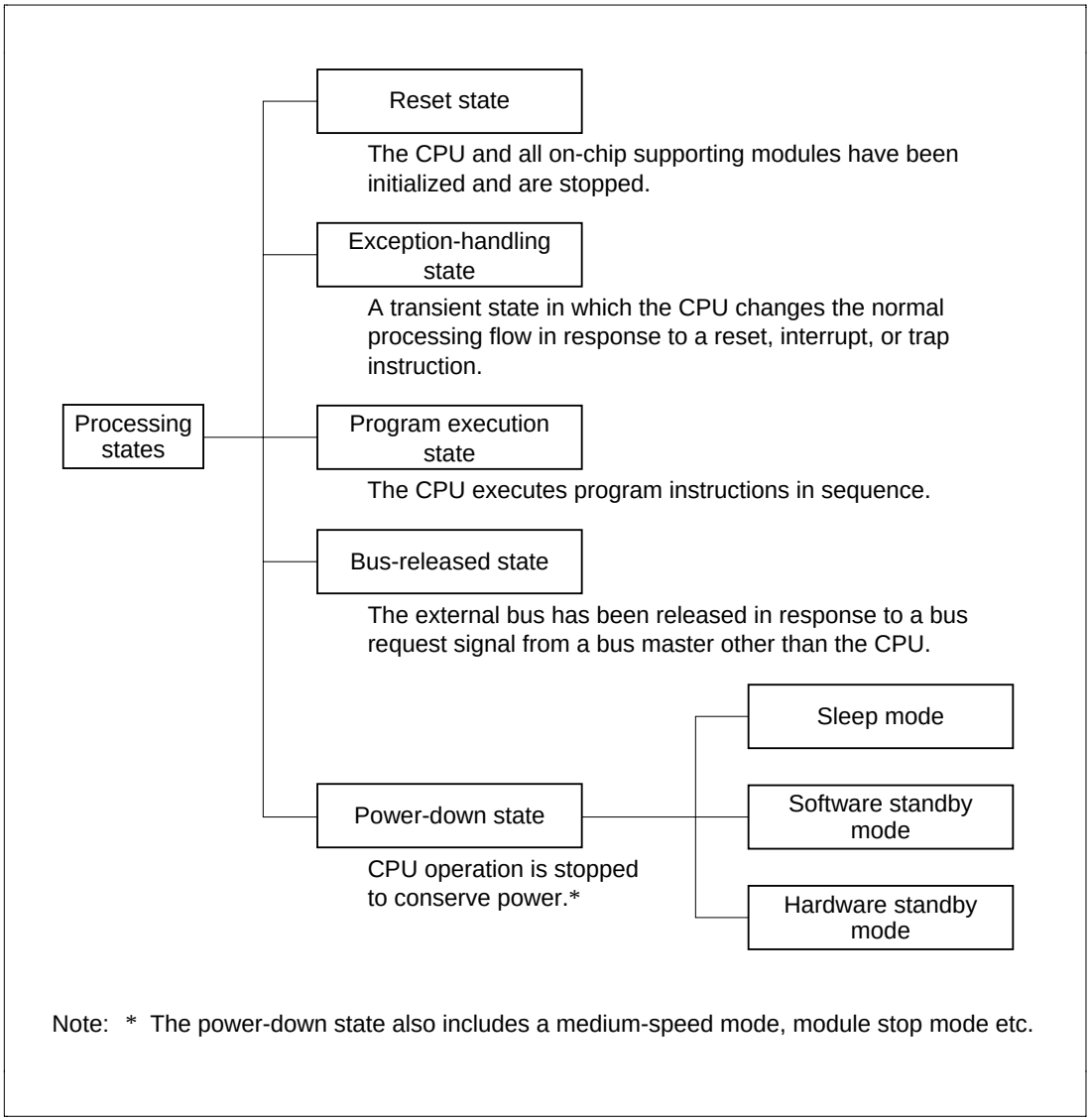
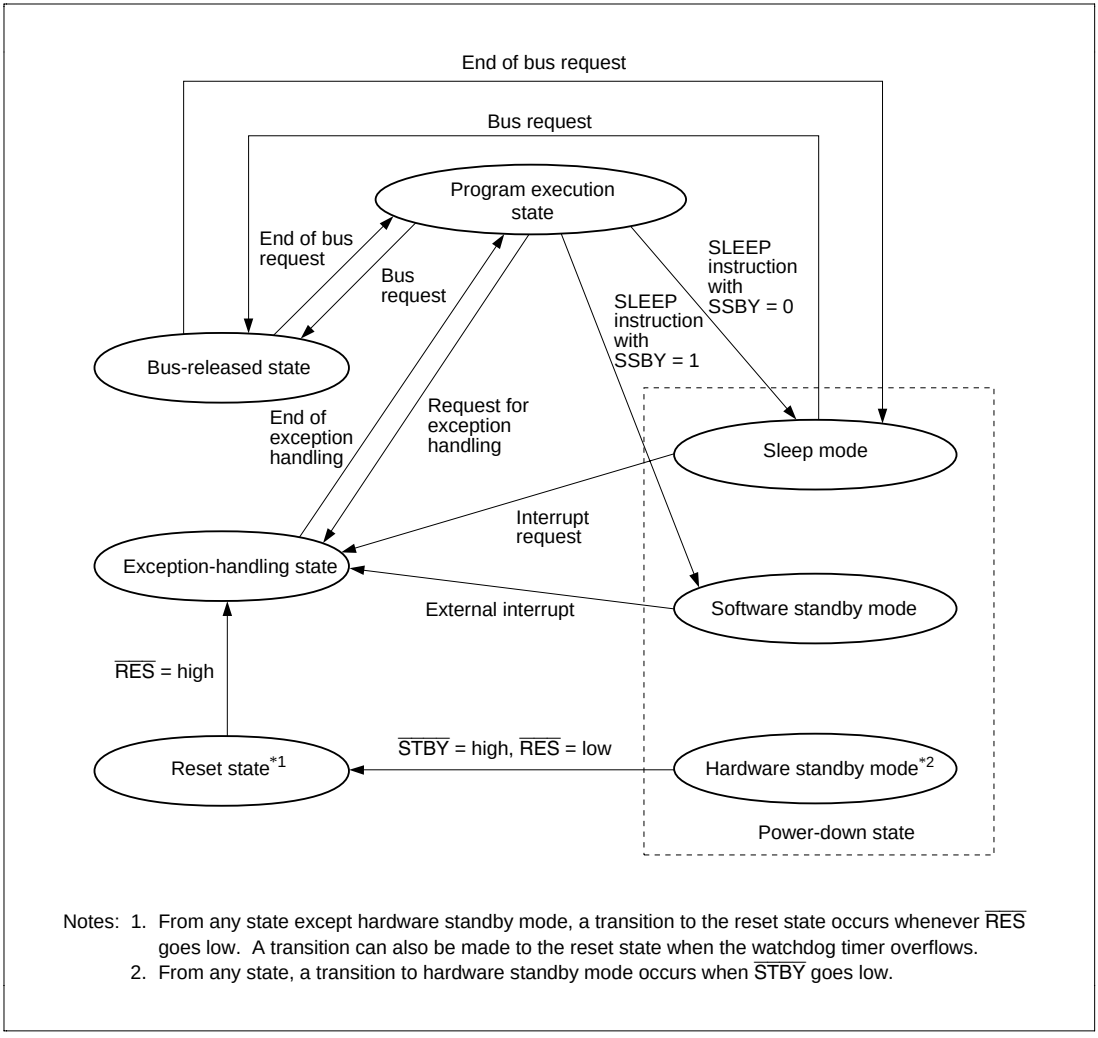



**Figure 2-14 Processing States**




Notes: 1. From any state except hardware standby mode, a transition to the reset state occurs whenever $\overline{RES}$ goes low. A transition can also be made to the reset state when the watchdog timer overflows.
2. From any state, a transition to hardware standby mode occurs when $\overline{STBY}$ goes low.

**Figure 2-15 State Transitions**

### 2.8.2 Reset State

When the $\overline{RES}$ input goes low all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{RES}$ signal changes from low to high.

The reset state can also be entered by a watchdog timer overflow. For details, refer to section 13, Watchdog Timer.

### 2.8.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to a reset, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address.

#### (1) Types of Exception Handling and Their Priority

Exception handling is performed for traces, resets, interrupts, and trap instructions. Table 2-4 indicates the types of exception handling and their priority. Trap instruction exception handling is always accepted, in the program execution state.

Exception handling and the stack structure depend on the interrupt control mode set in SYSCR.

**Table 2-4 Exception Handling Types and Priority**

| Priority | Type of Exception | Detection Timing | Start of Exception Handling |
|---|---|---|---|
| High ↑ Low | Reset | Synchronized with clock | Exception handling starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. |
| | Trace | End of instruction execution or end of exception-handling sequence*1 | When the trace (T) bit is set to 1, the trace starts at the end of the current instruction or current exception-handling sequence |
| | Interrupt | End of instruction execution or end of exception-handling sequence*2 | When an interrupt is requested, exception handling starts at the end of the current instruction or current exception-handling sequence |
| | Trap instruction | When TRAPA instruction is executed | Exception handling starts when a trap (TRAPA) instruction is executed*3 |

Notes: 1. Traces are enabled only in interrupt control modes 2 and 3. Trace exception-handling is not executed at the end of the RTE instruction.
2. Interrupts are not detected at the end of the ANDC, ORC, XORC, and LDC instructions, or immediately after reset exception handling.
3. Trap instruction exception handling is always accepted, in the program execution state.

### (2) Reset Exception Handling

After the $\overline{\text{RES}}$ pin has gone low and the reset state has been entered, when $\overline{\text{RES}}$ goes high again, reset exception handling starts. When reset exception handling starts the CPU fetches a start address (vector) from the exception vector table and starts program execution from that address. All interrupts, including NMI, are disabled during reset exception handling and after it ends.

### (3) Traces

Traces are enabled only in interrupt control modes 2 and 3. Trace mode is entered when the T bit of EXR is set to 1. When trace mode is established, trace exception handling starts at the end of each instruction.

At the end of a trace exception-handling sequence, the T bit of EXR is cleared to 0 and trace mode is cleared. Interrupt masks are not affected.

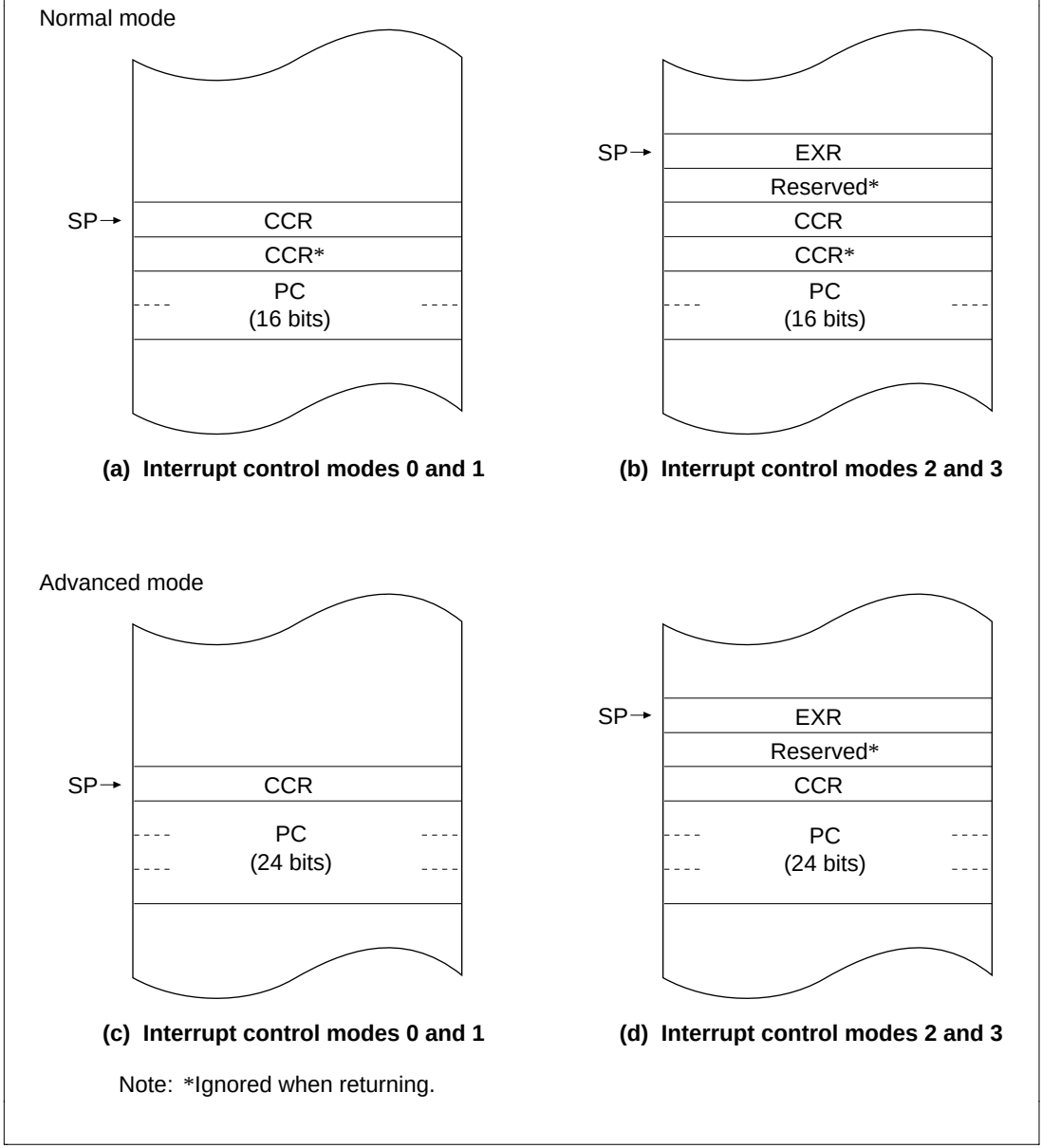
The T bit saved on the stack retains its value of 1, and when the RTE instruction is executed to return from the trace exception-handling routine, trace mode is entered again. Trace exception-handling is not executed at the end of the RTE instruction.

Trace mode is not entered in interrupt control modes 0 and 1, regardless of the state of the T bit.

### (4) Interrupt Exception Handling and Trap Instruction Exception Handling

When interrupt or trap-instruction exception handling begins, the CPU references the stack pointer (ER7) and pushes the program counter and other control registers onto the stack. Next, the CPU alters the settings of the interrupt mask bits in the control registers. Then the CPU fetches a start address (vector) from the exception vector table and program execution starts from that start address.

Figure 2-16 shows the stack after exception handling ends.




**(a) Interrupt control modes 0 and 1**

**(b) Interrupt control modes 2 and 3**

**(c) Interrupt control modes 0 and 1**

**(d) Interrupt control modes 2 and 3**

Note: *Ignored when returning.

**Figure 2-16 Stack Structure after Exception Handling (Examples)**

### 2.8.4 Program Execution State

In this state the CPU executes program instructions in sequence.

### 2.8.5 Bus-Released State

This is a state in which the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.

Bus masters other than the CPU are the direct memory access controller (DMAC) and data transfer controller (DTC).

For further details, refer to section 6, Bus Controller.

### 2.8.6 Power-Down State

The power-down state includes both modes in which the CPU stops operating and modes in which the CPU does not stop. There are three modes in which the CPU stops operating: sleep mode, software standby mode, and hardware standby mode. There are also two other power-down modes: medium-speed mode, and module stop mode. In medium-speed mode the CPU and other bus masters operate on a medium-speed clock. Module stop mode permits halting of the operation of individual modules, other than the CPU. For details, refer to section 21, Power-Down State.

**(1) Sleep Mode:** A transition to sleep mode is made if the SLEEP instruction is executed while the software standby bit (SSBY) in the standby control register (SBYCR) is cleared to 0. In sleep mode, CPU operations stop immediately after execution of the SLEEP instruction. The contents of CPU registers are retained.

**(2) Software Standby Mode:** A transition to software standby mode is made if the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1. In software standby mode, the CPU and clock halt and all MCU operations stop. As long as a specified voltage is supplied, the contents of CPU registers and on-chip RAM are retained. The I/O ports also remain in their existing states.

**(3) Hardware Standby Mode:** A transition to hardware standby mode is made when the $\overline{\text{STBY}}$ pin goes low. In hardware standby mode, the CPU and clock halt and all MCU operations stop. The on-chip supporting modules are reset, but as long as a specified voltage is supplied, on-chip RAM contents are retained.

## 2.9 Basic Timing

### 2.9.1 Overview

The H8S/2600 CPU is driven by a system clock, denoted by the symbol ø. The period from one rising edge of ø to the next is referred to as a "state." The memory cycle or bus cycle consists of one, two, or three states. Different methods are used to access on-chip memory, on-chip supporting modules, and the external address space.

### 2.9.2 On-Chip Memory (ROM, RAM)

On-chip memory is accessed in one state. The data bus is 16 bits wide, permitting both byte and word transfer instruction. Figure 2-17 shows the on-chip memory access cycle. Figure 2-18 shows the pin states.

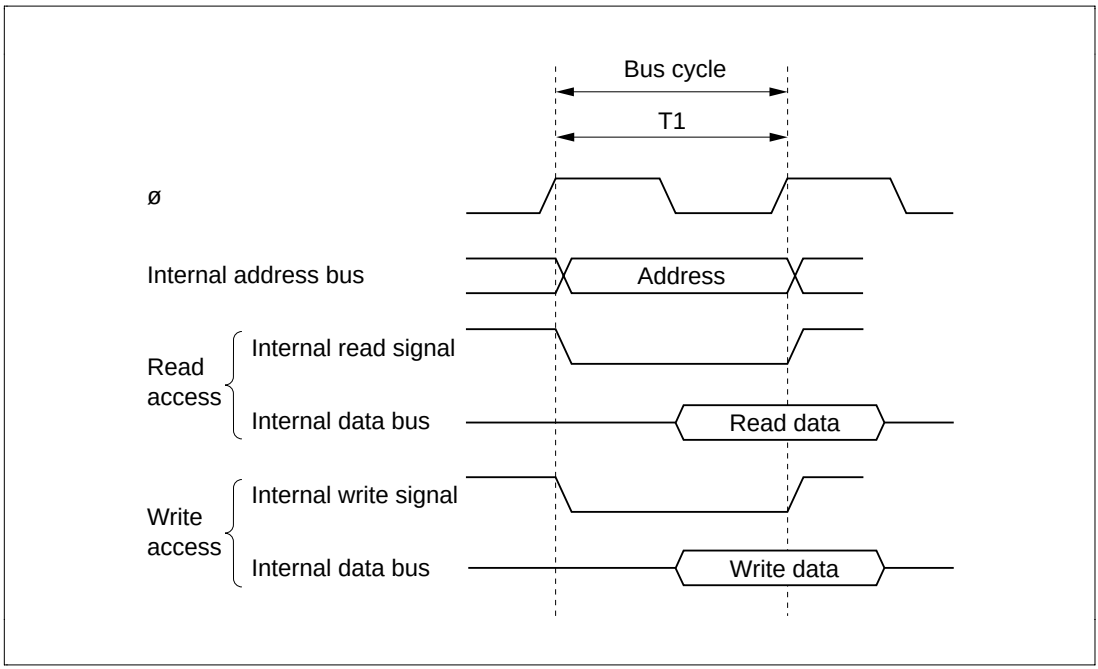



**Figure 2-17 On-Chip Memory Access Cycle**

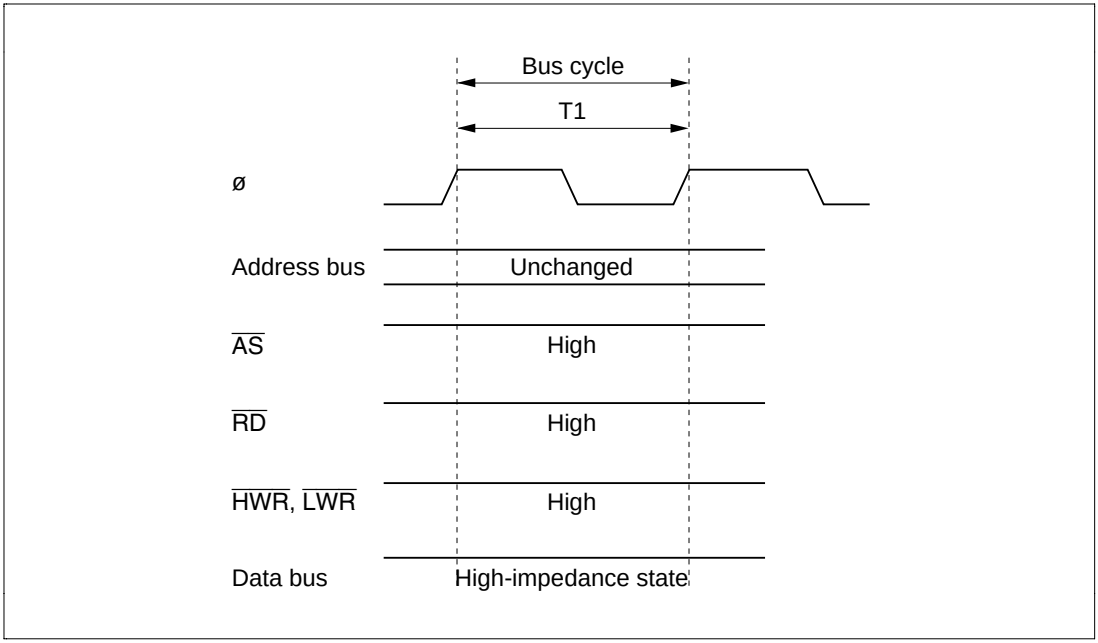



**Figure 2-18 Pin States during On-Chip Memory Access**

### 2.9.3 On-Chip Supporting Module Access Timing

The on-chip supporting modules are accessed in two states. The data bus is either 8 bits or 16 bits wide, depending on the particular internal I/O register being accessed. Figure 2-19 shows the access timing for the on-chip supporting modules. Figure 2-20 shows the pin states.

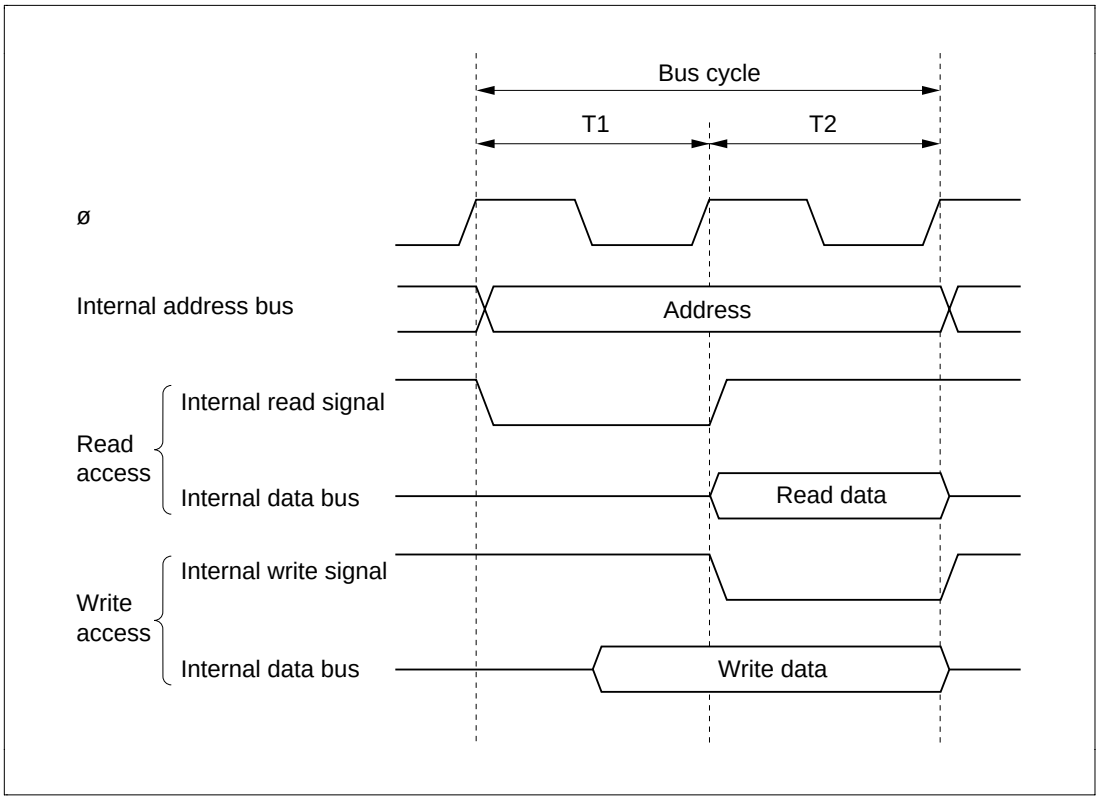



**Figure 2-19 On-Chip Supporting Module Access Cycle**

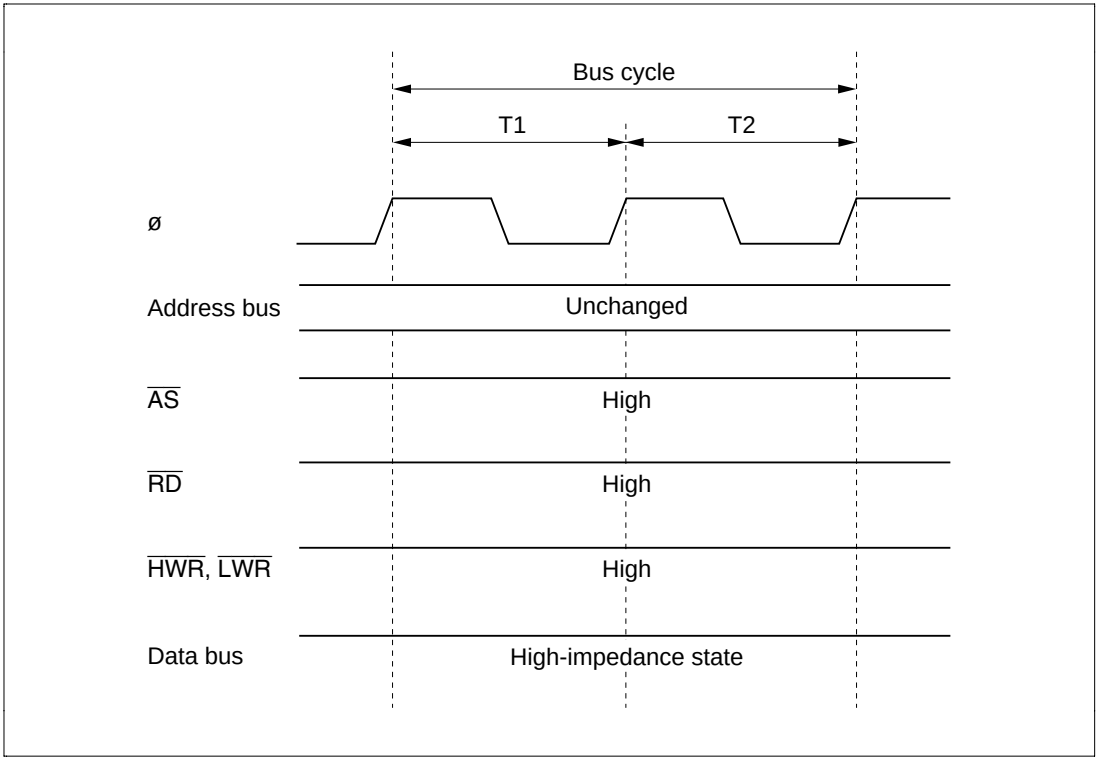



**Figure 2-20 Pin States during On-Chip Supporting Module Access**

### 2.9.4 External Address Space Access Timing

The external address space is accessed with an 8-bit or 16-bit data bus width in a two-state or three-state bus cycle. In three-state access, wait states can be inserted. For further details, refer to section 6, Bus Controller.

# Section 3 MCU Operating Modes

## 3.1 Overview

### 3.1.1 Operating Mode Selection

The H8S/2655 Series has seven operating modes (modes 1 to 7). These modes enable selection of the CPU operating mode, enabling/disabling of on-chip ROM, and the initial bus width setting, by setting the mode pins ($MD_2$ to $MD_0$).

Table 3-1 lists the MCU operating modes.

**Table 3-1 MCU Operating Mode Selection**

| MCU Operating Mode | $MD_2$ | $MD_1$ | $MD_0$ | CPU Operating Mode | Description | On-Chip ROM | External Data Bus: Initial Width | External Data Bus: Max. Width |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | — | — | — | — | |
| 1 | | | 1 | Normal | On-chip ROM disabled, expanded mode | Disabled | 8 bits | 16 bits |
| 2 | | 1 | 0 | | On-chip ROM enabled, expanded mode | Enabled | 8 bits | 16 bits |
| 3 | | | 1 | | Single-chip mode | | — | |
| 4 | 1 | 0 | 0 | Advanced | On-chip ROM disabled, expanded mode | Disabled | 16 bits | 16 bits |
| 5 | | | 1 | | | | 8 bits | 16 bits |
| 6 | | 1 | 0 | | On-chip ROM enabled, expanded mode | Enabled | 8 bits | 16 bits |
| 7 | | | 1 | | Single-chip mode | | — | |

The CPU's architecture allows for 4 Gbytes of address space, but the H8S/2655 Series actually accesses a maximum of 16 Mbytes.

Modes 1, 2, and 4 to 6 are externally expanded modes that allow access to external memory and peripheral devices.

The external expansion modes allow switching between 8-bit and 16-bit bus modes. After program execution starts, an 8-bit or 16-bit address space can be set for each area, depending on the bus controller setting. If 16-bit access is selected for any one area, 16-bit bus mode is set; if 8-bit access is selected for all areas, 8-bit bus mode is set.

Note that the functions of each pin depend on the operating mode.

The H8S/2655 Series can be used only in modes 1 to 7. This means that the mode pins must be set to select one of these modes. Do not change the inputs at the mode pins during operation.

### 3.1.2 Register Configuration

The H8S/2655 Series has a mode control register (MDCR) that indicates the inputs at the mode pins ($MD_2$ to $MD_0$), and a system control register (SYSCR) that controls the operation of the H8S/2655 Series. Table 3-2 summarizes these registers.

**Table 3-2 MCU Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Mode control register | MDCR | R | Undetermined | H'FF3B |
| System control register | SYSCR | R/W | H'01 | H'FF39 |

Note: * Lower 16 bits of the address.

## 3.2 Register Descriptions

### 3.2.1 Mode Control Register (MDCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | MDS2 | MDS1 | MDS0 |
| Initial value: | | 1 | 0 | 0 | 0 | 0 | —* | —* | —* |
| R/W | : | — | — | — | — | — | R | R | R |

Note: * Determined by pins $MD_2$ to $MD_0$.

MDCR is an 8-bit read-only register that indicates the current operating mode of the H8S/2655 Series.

**Bit 7—Reserved:** Read-only bit, always read as 1.

**Bits 6 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—Mode Select 2 to 0 (MDS2 to MDS0):** These bits indicate the input levels at pins $MD_2$ to $MD_0$ (the current operating mode). Bits MDS2 to MDS0 correspond to $MD_2$ to $MD_0$. MDS2 to MDS0 are read-only bits-they cannot be written to. The mode pin ($MD_2$ to $MD_0$) input levels are latched into these bits when MDCR is read. These latches are canceled by a power-on reset, but are retained after a manual reset.

### 3.2.2 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | MACS | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | — | R/W | R/W | R/W | — | — | R/W |

**Bit 7—MAC Saturation (MACS):** Selects either saturating or non-saturating calculation for the MAC instruction.

**Bit 7**

| MACS | Description | |
|---|---|---|
| 0 | Non-saturating calculation for MAC instruction | (Initial value) |
| 1 | Saturating calculation for MAC instruction | |

**Bit 6—Reserved:** Read-only bit, always read as 0.

**Bits 5 and 4—Interrupt Control Mode 1 and 0 (INTM1, INTM0):** These bits select the control mode of the interrupt controller. For details of the interrupt control modes, see section 5.4.1, Interrupt Control Modes and Interrupt Operation.

| Bit 5 | Bit 4 | Interrupt | | |
|---|---|---|---|---|
| INTM1 | INTM0 | Control Mode | Description | |
| 0 | 0 | 0 | Control of interrupts by I bit | (Initial value) |
| | 1 | 1 | Control of interrupts by I bit, U bit, and ICR | |
| 1 | 0 | 2 | Control of interrupts by I2 to I0 bits and IPR | |
| | 1 | 3 | Control of interrupts by I, UI, and I2 to I0 bits, and ICR and IPR | |

**Bit 3—NMI Edge Select (NMIEG):** Selects the valid edge of the NMI interrupt input.

| Bit 3 | | |
|---|---|---|
| NMIEG | Description | |
| 0 | An interrupt is requested at the falling edge of NMI input | (Initial value) |
| 1 | An interrupt is requested at the rising edge of NMI input | |

**Bits 2 and 1—Reserved:** Read-only bits, always read as 0.

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized when the reset status is released. It is not initialized in software standby mode.

| Bit 0 | | |
|---|---|---|
| RAME | Description | |
| 0 | On-chip RAM is disabled | |
| 1 | On-chip RAM is enabled | (Initial value) |

## 3.3 Operating Mode Descriptions

### 3.3.1 Mode 1

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is disabled, and 8-bit bus mode is set, immediately after a reset.

Ports B and C function as an address bus, port D functions as a data bus, and part of port F carries bus control signals. However, note that if 16-bit access is designated by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.2 Mode 2

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is enabled, and 8-bit bus mode is set. immediately after a reset.

Ports B and C function as input ports immediately after a reset. They can each be set to output addresses by setting the corresponding bits in the data direction register (DDR) to 1. Port D functions as a data bus, and part of port F carries bus control signals. However, note that if 16-bit access is designated by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

The amount of on-chip ROM that can be used is limited to 56 kbytes.

### 3.3.3 Mode 3

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

The amount of on-chip ROM that can be used is limited to 56 kbytes.

### 3.3.4 Mode 4

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Ports A, B and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 16 bits, with 16-bit access to all areas. However, note that if 8-bit access is designated by the bus controller for all areas, the bus mode switches to 8 bits.

### 3.3.5 Mode 5

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Ports A, B and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, note that if at least one area is designated for 16-bit access by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.6 Mode 6

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled.

Ports A, B and C function as input ports immediately after a reset. They can each be set to output addresses by setting the corresponding bits in the data direction register (DDR) to 1. Port D functions as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas.

### 3.3.7 Mode 7

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

## 3.4 Pin Functions in Each Operating Mode

The pin functions of ports A to F vary depending on the operating mode. Table 3-3 shows their functions in each operating mode.

**Table 3-3 Pin Functions in Each Mode**

| Port | | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|
| Port A | $PA_7$ to $PA_5$ | P | P | P | P*/A | P*/A | P*/A | P |
| | $PA_4$ to $PA_0$ | | | | A | A | | |
| Port B | | A | P*/A | P | A | A | P*/A | P |
| Port C | | A | P*/A | P | A | A | P*/A | P |
| Port D | | D | D | P | D | D | D | P |
| Port E | | P*/D | P*/D | P | P*/D | P*/D | P*/D | P |
| Port F | $PF_7$ | P*/C* | P*/C* | P*/C | P*/C* | P*/C* | P*/C* | P*/C |
| | $PF_6$ to $PF_3$ | C | C | P | C | C | C | P |
| | $PF_2$ to $PF_0$ | P*/C | P*/C | | P*/C | P*/C | P*/C | |

**Legend**

P: I/O port

A: Address bus output

D: Data bus I/O

C: Control signals, clock I/O

*: After reset

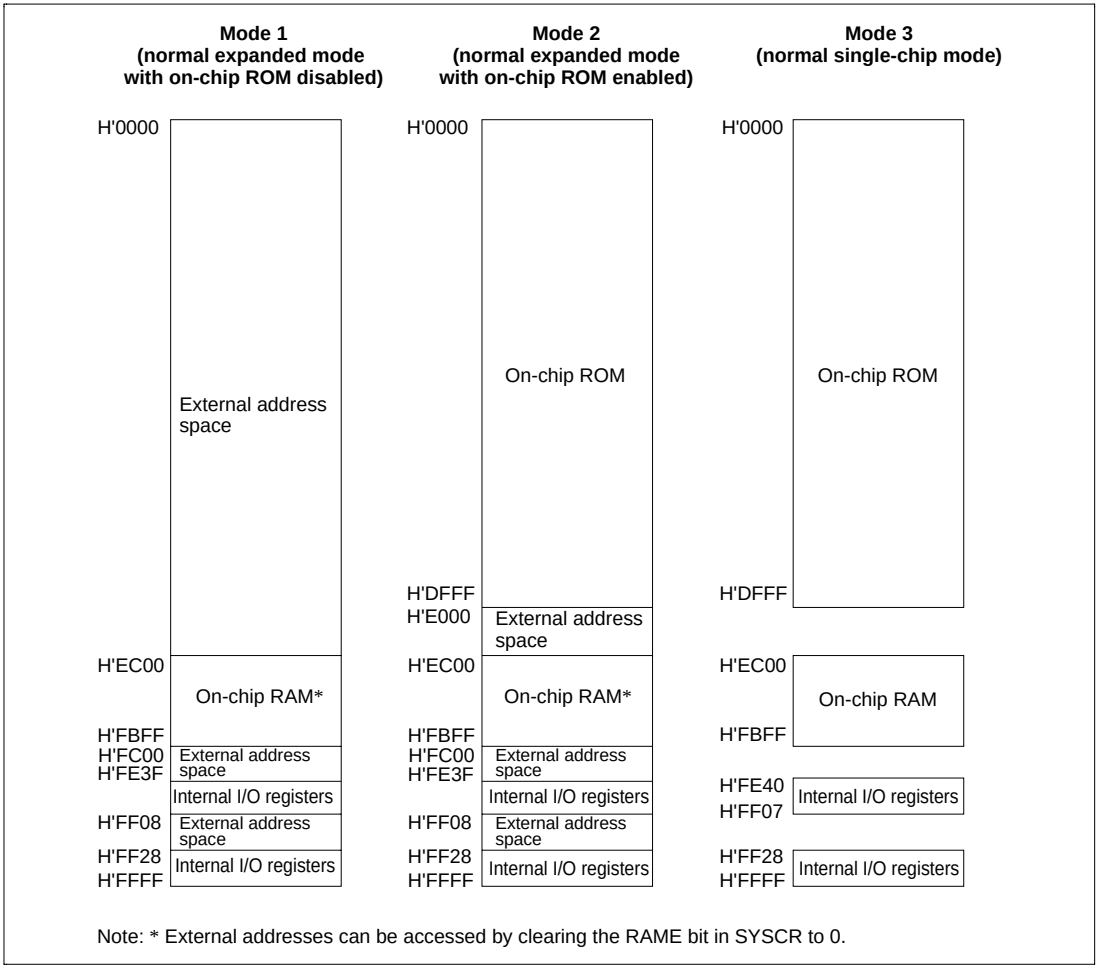
## 3.5 Memory Map in Each Operating Mode

Figure 3-1 shows a memory map for each of the operating modes.

The address space is 64 kbytes in modes 1 to 3 (normal modes), and 16 Mbytes in modes 4 to 7 (advanced modes).

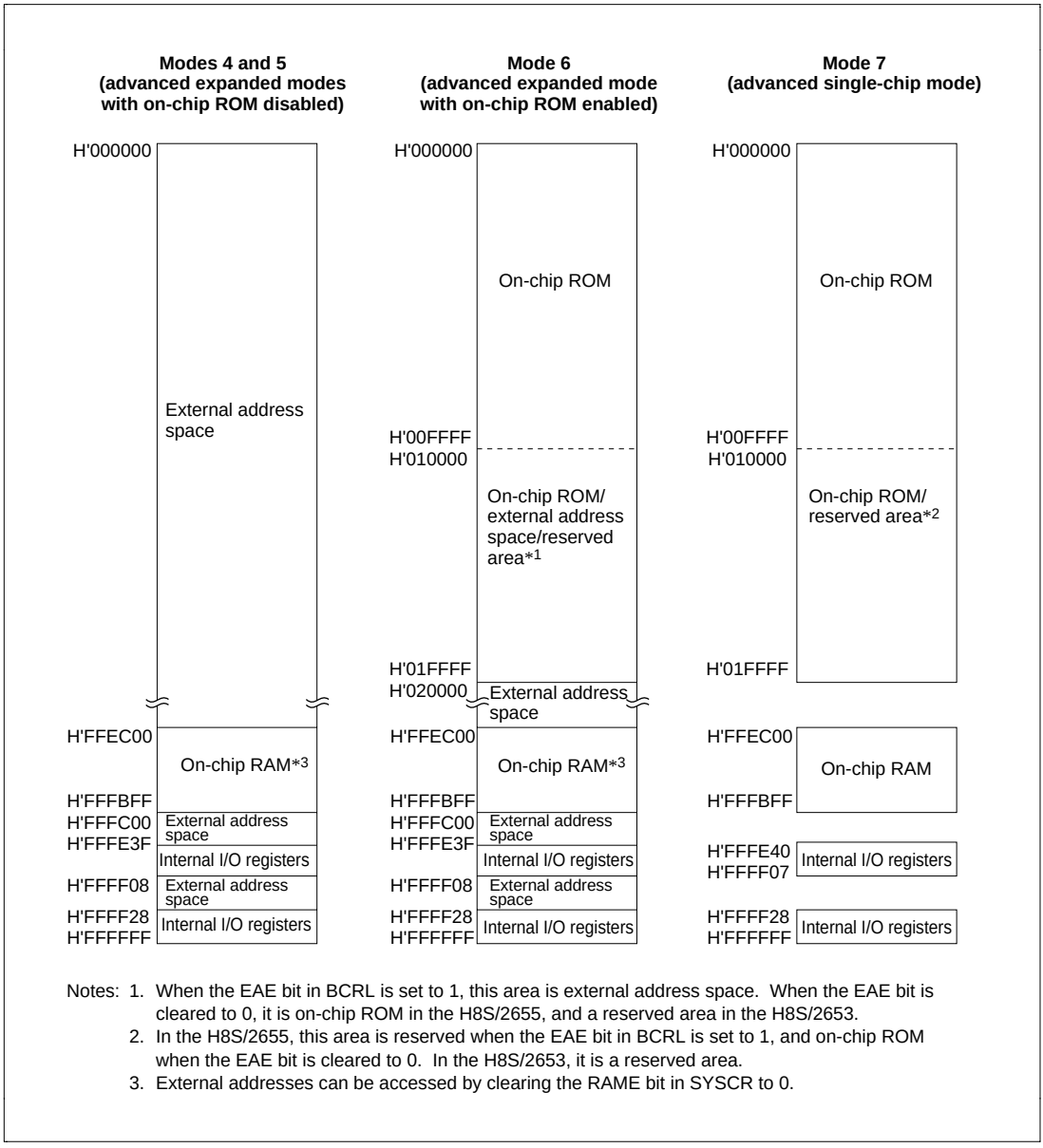
The on-chip ROM of H8S/2655 contains 128 kbytes, but only 56 kbytes are available in modes 2 and 3 (normal modes).

The address space is divided into eight areas for modes 4 to 7. For details, see section 6, Bus Controller.




Note: * External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3-1 Memory Map in Each Operating Mode**




Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space. When the EAE bit is cleared to 0, it is on-chip ROM in the H8S/2655, and a reserved area in the H8S/2653.
2. In the H8S/2655, this area is reserved when the EAE bit in BCRL is set to 1, and on-chip ROM when the EAE bit is cleared to 0. In the H8S/2653, it is a reserved area.
3. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3-1 Memory Map in Each Operating Mode (cont)**

# Section 4 Exception Handling

## 4.1 Overview

### 4.1.1 Exception Handling Types and Priority

As table 4-1 indicates, exception handling may be caused by a reset, trap instruction, or interrupt. Exception handling is prioritized as shown in table 4-1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Trap instruction exceptions are accepted at all times, in the program execution state. See appendix D.1, Port States in Each Mode.

Exception handling sources, the stack structure, and the operation of the CPU vary depending on the interrupt control mode set by the INTM0 and INTM1 bits of SYSCR.

**Table 4-1 Exception Types and Priority**

| Priority | Exception Type | Start of Exception Handling |
|---|---|---|
| High ↑ | Reset | Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. |
| | Trace*1 | Starts when execution of the current instruction or exception handling ends, if the trace (T) bit is set to 1 |
| | Interrupt | Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued*2 |
| Low | Trap instruction (TRAPA)*3 | Started by execution of a trap instruction (TRAPA) |

Notes: 1. Traces are enabled only in interrupt control modes 2 and 3. Trace exception handling is not executed after execution of an RTE instruction.
2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
3. Trap instruction exception handling requests are accepted at all times in program execution state.

### 4.1.2 Exception Handling Operation

Exceptions originate from various sources. Trap instructions and interrupts are handled as follows:

1. The program counter (PC), condition code register (CCR), and extended register (EXR) are pushed onto the stack.
2. The interrupt mask bits are updated. The T bit is cleared to 0.
3. A vector address corresponding to the exception source is generated, and program execution starts from that address.

For a reset exception, steps 2 and 3 above are carried out.

### 4.1.3 Exception Vector Table

The exception sources are classified as shown in figure 4-1. Different vector addresses are assigned to different exception sources.

Table 4-2 lists the exception sources and their vector addresses.

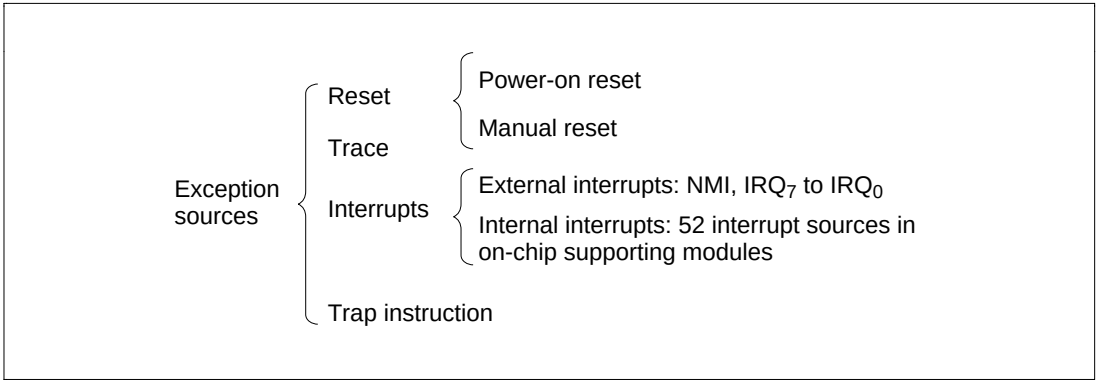



**Figure 4-1 Exception Sources**

In modes 6 and 7 in the H8S/2655, the on-chip ROM available for use after a power-on reset is the 64-kbyte area comprising addresses H'000000 to H'00FFFF. Care is required when setting vector addresses. In this case, clearing the EAE bit in BCRL enables the 128-kbyte area comprising addresses H'000000 to H'01FFFF to be used.

**Table 4-2 Exception Vector Table**

| Exception Source | | Vector Number | Vector Address*[1] Normal Mode | Vector Address*[1] Advanced Mode |
|---|---|---|---|---|
| Power-on reset | | 0 | H'0000 to H'0001 | H'0000 to H'0003 |
| Manual reset | | 1 | H'0002 to H'0003 | H'0004 to H'0007 |
| Reserved for system use | | 2 | H'0004 to H'0006 | H'0008 to H'000B |
| | | 3 | H'0006 to H'0007 | H'000C to H'000F |
| | | 4 | H'0008 to H'0009 | H'0010 to H'0013 |
| Trace | | 5 | H'000A to H'000B | H'0014 to H'0017 |
| Reserved for system use | | 6 | H'000C to H'000D | H'0018 to H'001B |
| External interrupt | NMI | 7 | H'000E to H'000F | H'001C to H'001F |
| Trap instruction (4 sources) | | 8 | H'0010 to H'0011 | H'0020 to H'0023 |
| | | 9 | H'0012 to H'0013 | H'0024 to H'0027 |
| | | 10 | H'0014 to H'0015 | H'0028 to H'002B |
| | | 11 | H'0016 to H'0017 | H'002C to H'002F |
| Reserved for system use | | 12 | H'0018 to H'0019 | H'0030 to H'0033 |
| | | 13 | H'001A to H'001B | H'0034 to H'0037 |
| | | 14 | H'001C to H'001D | H'0038 to H'003B |
| | | 15 | H'001E to H'001F | H'003C to H'003F |
| External interrupt | $IRQ_0$ | 16 | H'0020 to H'0021 | H'0040 to H'0043 |
| | $IRQ_1$ | 17 | H'0022 to H'0023 | H'0044 to H'0047 |
| | $IRQ_2$ | 18 | H'0024 to H'0025 | H'0048 to H'004B |
| | $IRQ_3$ | 19 | H'0026 to H'0027 | H'004C to H'004F |
| | $IRQ_4$ | 20 | H'0028 to H'0029 | H'0050 to H'0053 |
| | $IRQ_5$ | 21 | H'002A to H'002B | H'0054 to H'0057 |
| | $IRQ_6$ | 22 | H'002C to H'002D | H'0058 to H'005B |
| | $IRQ_7$ | 23 | H'002E to H'002F | H'005C to H'005F |
| Internal interrupt*[2] | | 24<br>⋮<br>91 | H'0030 to H'0031<br>⋮<br>H'00B6 to H'00B7 | H'0060 to H'0063<br>⋮<br>H'016C to H'016F |

Notes: 1. Lower 16 bits of the address.

2. For details of internal interrupt vectors, see section 5.3.3, Interrupt Exception Handling Vector Table.

## 4.2 Reset

### 4.2.1 Overview

A reset has the highest exception priority.

When the $\overline{\text{RES}}$ pin goes low, all processing halts and the H8S/2655 Series enters the reset state. A reset initializes the internal state of the CPU and the registers of on-chip supporting modules. Immediately after a reset, interrupt control mode 0 is set.

Reset exception handling begins when the $\overline{\text{RES}}$ pin changes from low to high.

The level of the NMI pin at reset determines whether the type of reset is a power-on reset or a manual reset.

The H8S/2655 Series can also be reset by overflow of the watchdog timer. For details see section 13, Watchdog Timer.

### 4.2.2 Reset Types

A reset can be of either of two types: a power-on reset or a manual reset. Reset types are shown in table 4-3. A power-on reset should be used when powering on.

The internal state of the CPU is initialized by either type of reset. A power-on reset also initializes all the registers in the on-chip supporting modules, while a manual reset initializes all the registers in the on-chip supporting modules except for the bus controller and I/O ports, which retain their previous states.
With a manual reset, since the on-chip supporting modules are initialized, ports used as on-chip supporting module I/O pins are switched to I/O ports controlled by DDR and DR.

**Table 4-3 Reset Types**

| | Reset Transition Conditions | | Internal State | |
|---|---|---|---|---|
| **Type** | **NMI** | **$\overline{\text{RES}}$** | **CPU** | **On-Chip Supporting Modules** |
| Power-on reset | High | Low | Initialized | Initialized |
| Manual reset | Low | Low | Initialized | Initialized, except for bus controller and I/O ports |

A reset caused by the watchdog timer can also be of either of two types: a power-on reset or a manual reset.

### 4.2.3 Reset Sequence

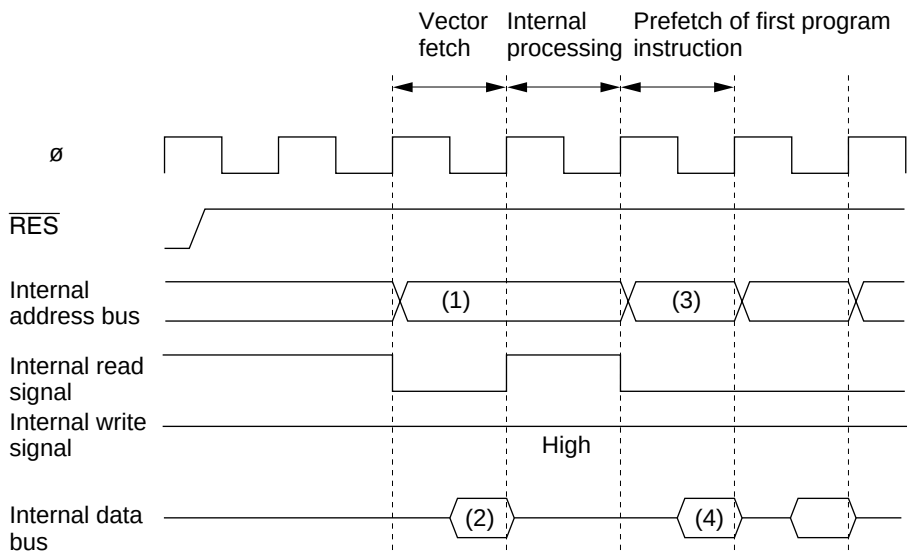
The H8S/2655 Series enters the reset state when the $\overline{\text{RES}}$ pin goes low.

To ensure that the H8S/2655 Series is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-up. To reset the H8S/2655 Series during operation, hold the $\overline{\text{RES}}$ pin low for at least 20 states.

When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, the H8S/2655 Series starts reset exception handling as follows:

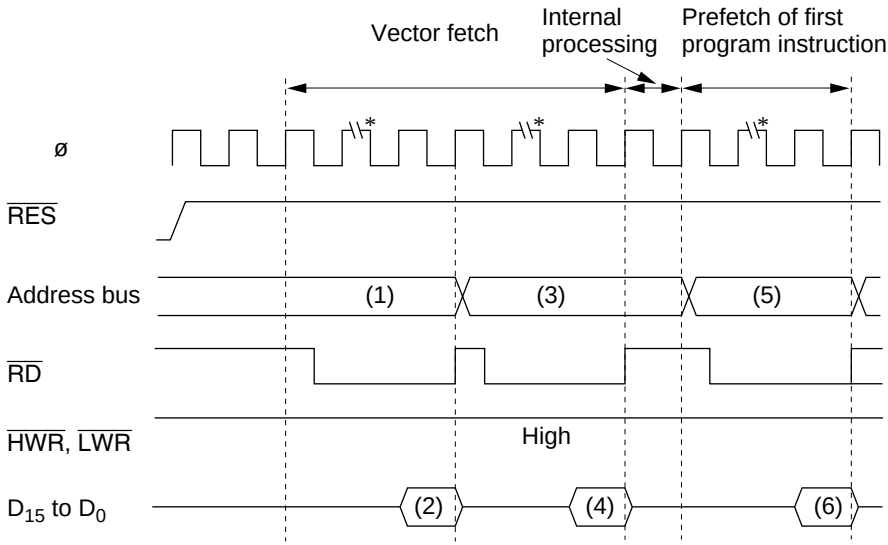
1. The internal state of the CPU and the registers of the on-chip supporting modules are initialized, and the I bit is set to 1 in CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figures 4-2 and 4-3 show examples of the reset sequence.




(1) Reset exception handling vector address ((1) = H'0000)
(2) Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2))
(4) First program instruction

**Figure 4-2 Reset Sequence (Modes 2 and 3)**




(1) (3) Reset exception handling vector address ((1) = H'000000, (3) = H'000002)
(2) (4) Start address (contents of reset exception handling vector address)
(5) Start address ((5) = (2) (4))
(6) First program instruction

Note: * 3 program wait states are inserted.

**Figure 4-3 Reset Sequence (Mode 4)**

### 4.2.4 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.L #xx:32, SP).

## 4.3 Traces

Traces are enabled in interrupt control modes 2 and 3. Trace mode is not activated in interrupt control modes 0 and 1, irrespective of the state of the T bit. For details of interrupt control modes, see section 5, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction.

Trace mode is canceled by clearing the T bit in EXR to 0. It is not affected by interrupt masking.

Table 4-4 shows the state of CCR and EXR after execution of trace exception handling.

Interrupts are accepted even within the trace exception handling routine.

The T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes.

Trace exception handling is not carried out after execution of the RTE instruction.

**Table 4-4 Status of CCR and EXR after Trace Exception Handling**

| | CCR | | EXR | |
|---|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **I2 to I0** | **T** |
| 0 | Trace exception handling cannot be used. | | | |
| 1 | | | | |
| 2 | 1 | — | — | 0 |
| 3 | 1 | 1 | — | 0 |

**Legend**

1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

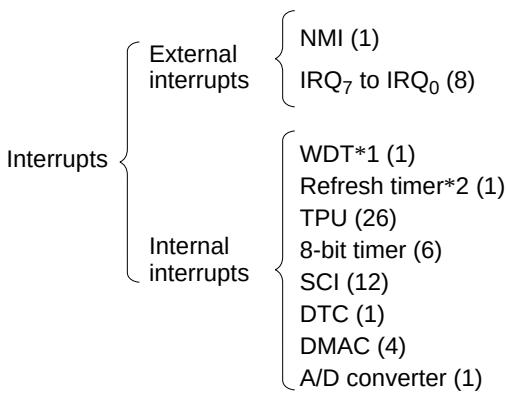
## 4.4 Interrupts

Interrupt exception handling can be requested by nine external sources (NMI, $IRQ_7$ to $IRQ_0$) and 52 internal sources in the on-chip supporting modules. Figure 4-4 classifies the interrupt sources and the number of interrupts of each type.

The on-chip supporting modules that can request interrupts include the watchdog timer (WDT), refresh timer, 16-bit timer-pulse unit (TPU), 8-bit timer, serial communication interface (SCI), data transfer controller (DTC), DMA controller (DMAC), and A/D converter. Each interrupt source has a separate vector address.

NMI is the highest-priority interrupt, and is always accepted. Interrupts are controlled by the interrupt controller. The interrupt controller has four interrupt control modes and can assign interrupts other than NMI to either three or eight priority/mask levels to enable multiplexed interrupt control.

For details of interrupts, see section 5, Interrupt Controller.




Notes: Numbers in parentheses are the numbers of interrupt sources.
1. When the watchdog timer is used as an interval timer, it generates an interrupt request at each counter overflow.
2. When the refresh timer is used as an interval timer, it generates an interrupt request at each compare match.

**Figure 4-4 Interrupt Sources and Number of Interrupts**

## 4.5 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state.

The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4-5 shows the status of CCR and EXR after execution of trap instruction exception handling.

**Table 4-5 Status of CCR and EXR after Trap Instruction Exception Handling**

| | CCR | | EXR | |
|---|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **I2 to I0** | **T** |
| 0 | 1 | — | — | — |
| 1 | 1 | 1 | — | — |
| 2 | 1 | — | — | 0 |
| 3 | 1 | 1 | — | 0 |

**Legend**

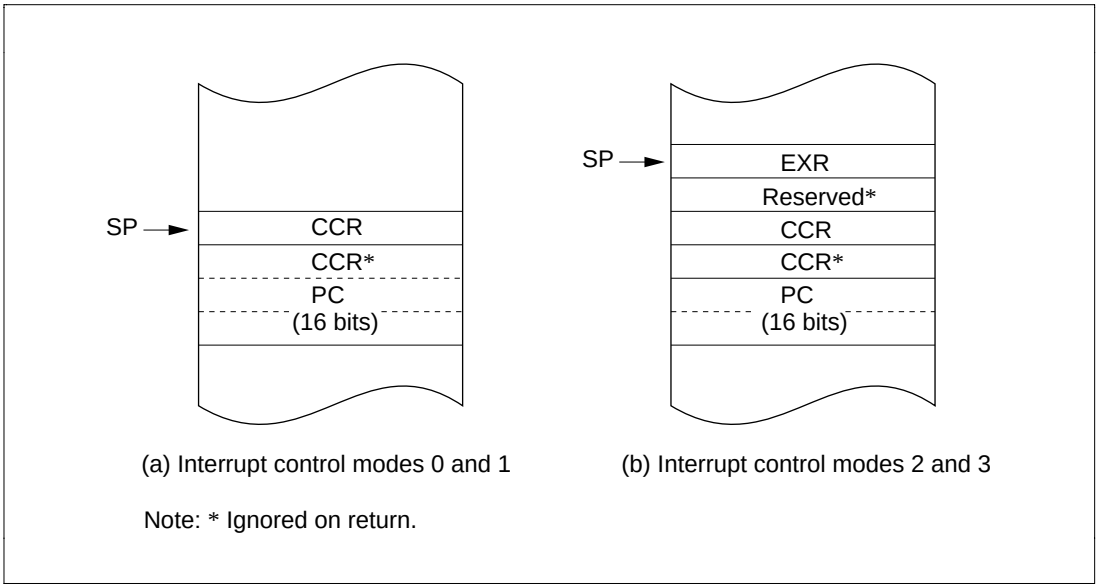
1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

## 4.6 Stack Status after Exception Handling

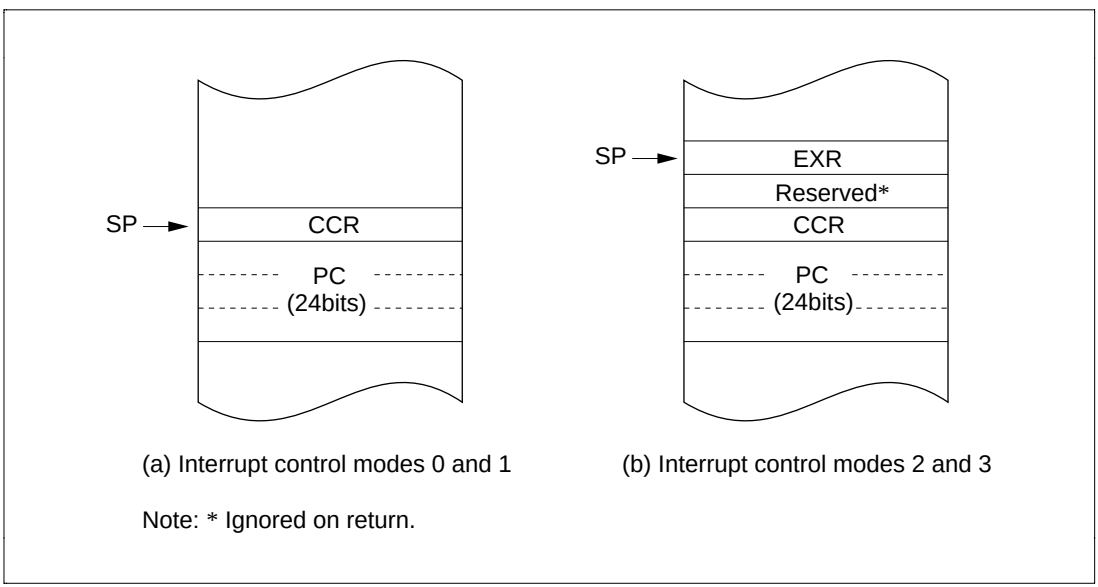
Figure 4-5 shows the stack after completion of trap instruction exception handling and interrupt exception handling.




(a) Interrupt control modes 0 and 1 (b) Interrupt control modes 2 and 3

Note: * Ignored on return.

**Figure 4-5 (1) Stack Status after Exception Handling (Normal Modes)**




(a) Interrupt control modes 0 and 1 (b) Interrupt control modes 2 and 3

Note: * Ignored on return.

**Figure 4-5 (2) Stack Status after Exception Handling (Advanced Modes)**

## 4.7 Notes on Use of the Stack

When accessing word data or longword data, the H8S/2655 Series assumes that the lowest address bit is 0. The stack should always be accessed by word transfer instruction or longword transfer instruction, and the value of the stack pointer (SP, ER7) should always be kept even. Use the following instructions to save registers:

```
PUSH.W   Rn    (or MOV.W Rn, @-SP)
PUSH.L   ERn   (or MOV.L ERn, @-SP)
```

Use the following instructions to restore registers:

```
POP.W    Rn    (or MOV.W @SP+, Rn)
POP.L    ERn   (or MOV.L @SP+, ERn)
```

Setting SP to an odd value may lead to a malfunction. Figure 4-6 shows an example of what happens when the SP value is odd.

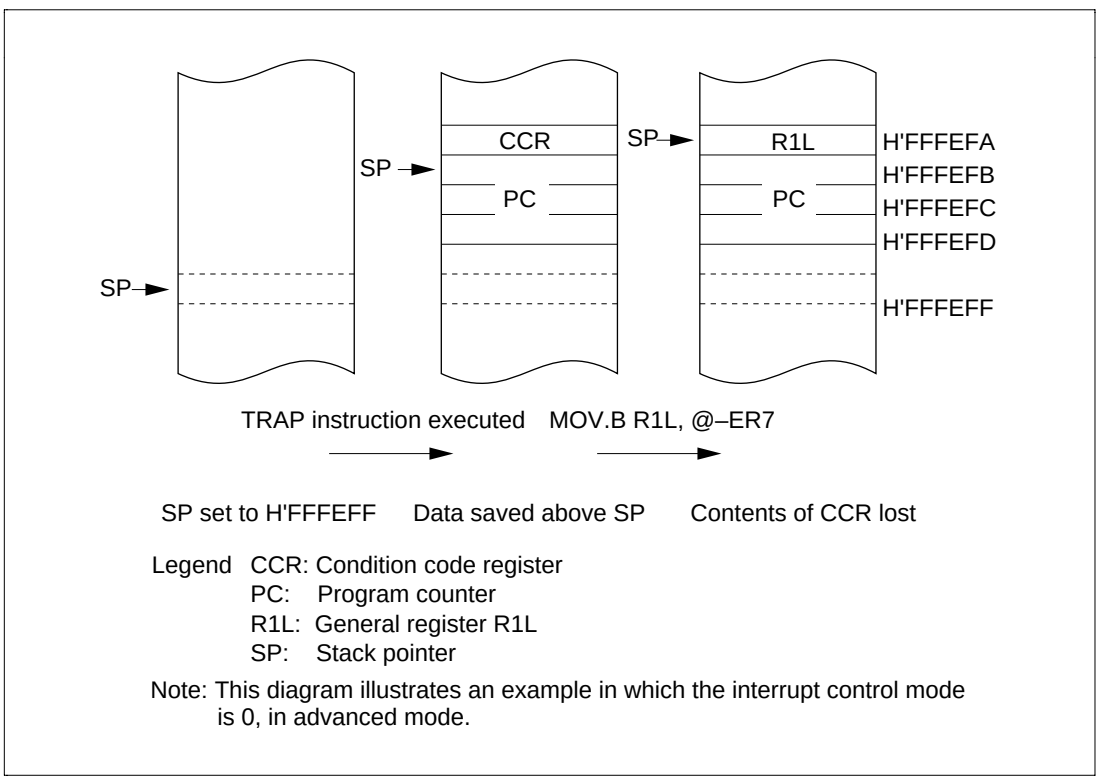



**Figure 4-6 Operation when SP Value is Odd**

# Section 5 Interrupt Controller

## 5.1 Overview

### 5.1.1 Features

The H8S/2655 Series controls interrupts by means of an interrupt controller. The interrupt controller has the following features:

- Four interrupt control modes
  - — Any of four interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).
- Priorities settable with ICR
  - — An interrupt control register (ICR) is provided for setting interrupt priorities. Three priority levels can be set for each module for all interrupts except NMI.
- Priorities settable with IPR
  - — An interrupt priority register (IPR) is provided for setting interrupt priorities. Eight priority levels can be set for each module for all interrupts except NMI.
  - — NMI is assigned the highest priority level of 8, and can be accepted at all times.
- Independent vector addresses
  - — All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- Nine external interrupts
  - — NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge can be selected for NMI.
  - — Falling edge, rising edge, or both edge detection, or level sensing, can be selected for IRQ7 to IRQ0.
- DTC and DMAC control
  - — DTC and DMAC activation is performed by means of interrupts.

### 5.1.2 Block Diagram

A block diagram of the interrupt controller is shown in Figure 5-1.

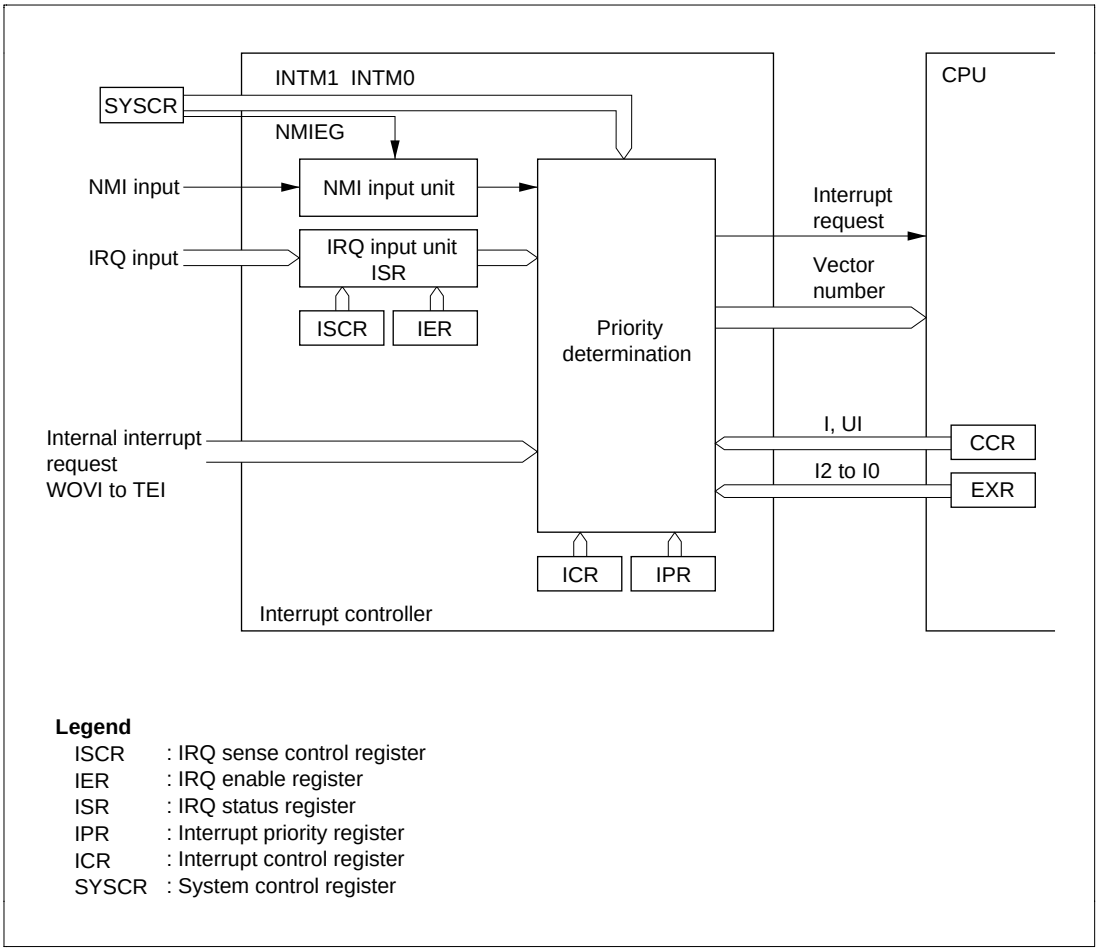



**Figure 5-1 Block Diagram of Interrupt Controller**

### 5.1.3 Pin Configuration

Table 5-1 summarizes the pins of the interrupt controller.

**Table 5-1 Interrupt Controller Pins**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Nonmaskable interrupt | NMI | Input | Nonmaskable external interrupt; rising or falling edge can be selected |
| External interrupt requests 7 to 0 | $\overline{IRQ}_7$ to $\overline{IRQ}_0$ | Input | Maskable external interrupts; rising, falling, or both edges, or level sensing, can be selected |

### 5.1.4 Register Configuration

Table 5-2 summarizes the registers of the interrupt controller.

**Table 5-2 Interrupt Controller Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| System control register | SYSCR | R/W | H'01 | H'FF39 |
| IRQ sense control register H | ISCRH | R/W | H'00 | H'FF2C |
| IRQ sense control register L | ISCRL | R/W | H'00 | H'FF2D |
| IRQ enable register | IER | R/W | H'00 | H'FF2E |
| IRQ status register | ISR | R/(W)*[2] | H'00 | H'FF2F |
| Interrupt control register A | ICRA | R/W | H'00 | H'FEC0 |
| Interrupt control register B | ICRB | R/W | H'00 | H'FEC1 |
| Interrupt control register C | ICRC | R/W | H'00 | H'FEC2 |
| Interrupt priority register A | IPRA | R/W | H'77 | H'FEC4 |
| Interrupt priority register B | IPRB | R/W | H'77 | H'FEC5 |
| Interrupt priority register C | IPRC | R/W | H'77 | H'FEC6 |
| Interrupt priority register D | IPRD | R/W | H'77 | H'FEC7 |
| Interrupt priority register E | IPRE | R/W | H'77 | H'FEC8 |
| Interrupt priority register F | IPRF | R/W | H'77 | H'FEC9 |
| Interrupt priority register G | IPRG | R/W | H'77 | H'FECA |
| Interrupt priority register H | IPRH | R/W | H'77 | H'FECB |
| Interrupt priority register I | IPRI | R/W | H'77 | H'FECC |
| Interrupt priority register J | IPRJ | R/W | H'77 | H'FECD |
| Interrupt priority register K | IPRK | R/W | H'77 | H'FECE |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 5.2 Register Descriptions

### 5.2.1 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | MACS | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | — | R/W | R/W | R/W | — | — | R/W |

SYSCR is an 8-bit readable/writable register that selects the interrupt control mode, and the detected edge for NMI.

Only bits 5 to 3 are described here; for details of the other bits, see section 3.2.2, System Control Register (SYSCR).

SYSCR is initialized to H'01 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 5 and 4—Interrupt Control Mode 1 and 0 (INTM1, INTM0):** These bits select one of four interrupt control modes for the interrupt controller.

| Bit 5 | Bit 4 | Interrupt | | |
|---|---|---|---|---|
| **INTM1** | **INTM0** | **Control Mode** | **Description** | |
| 0 | 0 | 0 | Interrupts are controlled by I bit | (Initial value) |
| | 1 | 1 | Interrupts are controlled by I and UI bits and ICR | |
| 1 | 0 | 2 | Interrupts are controlled by bits I2 to I0, and IPR | |
| | 1 | 3 | Interrupts are controlled by bits I, UI, and I2 to I0, ICR, and IPR | |

**Bit 3—NMI Edge Select (NMIEG):** Selects the input edge for the NMI pin.

| Bit 3 | | |
|---|---|---|
| **NMIEG** | **Description** | |
| 0 | Interrupt request generated at falling edge of NMI input | (Initial value) |
| 1 | Interrupt request generated at rising edge of NMI input | |

### 5.2.2 Interrupt Control Registers A to C (ICRA to ICRC)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The ICR registers are three 8-bit readable/writable registers that set the interrupt control level for interrupts other than NMI.

The correspondence between ICR settings and interrupt sources is shown in table 5-3.

The ICR registers are initialized to H'00 by a reset and in hardware standby mode.

**Table 5-3 Correspondence between Interrupt Sources and ICR Settings**

| | Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Register** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| ICRA | $IRQ_0$ | $IRQ_1$ | $IRQ_2$<br>$IRQ_3$ | $IRQ_4$<br>$IRQ_5$ | $IRQ_6$<br>$IRQ_7$ | DTC | Watchdog timer | Refresh timer |
| ICRB | — | A/D converter | TPU channel 0 | TPU channel 1 | TPU channel 2 | TPU channel 3 | TPU channel 4 | TPU channel 5 |
| ICRC | 8-bit timer channel 0 | 8-bit timer channel 1 | DMAC | SCI channel 0 | SCI channel 1 | SCI channel 2 | — | — |

### 5.2.3 Interrupt Priority Registers A to K (IPRA to IPRK)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 |
| Initial value: | | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W | : | — | R/W | R/W | R/W | — | R/W | R/W | R/W |

The IPR registers are eleven 8-bit readable/writable registers that set priorities (levels 7 to 0) for interrupts other than NMI.

The correspondence between IPR settings and interrupt sources is shown in table 5-4.

The IPR registers set a priority (level 7 to 0) for each interrupt source other than NMI.

The IPR registers are initialized to H'77 by a reset and in hardware standby mode.

**Bits 7 and 3—Reserved:** Read-only bits, always read as 0.

**Table 5-4 Correspondence between Interrupt Sources and IPR Settings**

| | Bits | |
|---|---|---|
| **Register** | **6 to 4** | **2 to 0** |
| IPRA | $IRQ_0$ | $IRQ_1$ |
| IPRB | $IRQ_2$<br>$IRQ_3$ | $IRQ_4$<br>$IRQ_5$ |
| IPRC | $IRQ_6$<br>$IRQ_7$ | DTC |
| IPRD | Watchdog timer | Refresh timer |
| IPRE | — | A/D converter |
| IPRF | TPU channel 0 | TPU channel 1 |
| IPRG | TPU channel 2 | TPU channel 3 |
| IPRH | TPU channel 4 | TPU channel 5 |
| IPRI | 8-bit timer channel 0 | 8-bit timer channel 1 |
| IPRJ | DMAC | SCI channel 0 |
| IPRK | SCI channel 1 | SCI channel 2 |

As shown in table 5-4, multiple interrupts are assigned to one IPR. Setting a value in the range from H'0 to H'7 in the 3-bit groups of bits 6 to 4 and 2 to 0 sets the priority of the corresponding interrupt. The lowest priority level, level 0, is assigned by setting H'0, and the highest priority level, level 7, by setting H'7.

When interrupt requests are generated, the highest-priority interrupt according to the priority levels set in the IPR registers is selected. This interrupt level is then compared with the interrupt mask level set by the interrupt mask bits (I2 to I0) in the extend register (EXR) in the CPU, and if the priority level of the interrupt is higher than the set mask level, an interrupt request is issued to the CPU.

### 5.2.4 IRQ Enable Register (IER)

IER is an 8-bit readable/writable register that controls enabling and disabling of interrupt requests $IRQ_7$ to $IRQ_0$.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IER is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—$IRQ_7$ to $IRQ_0$ Enable (IRQ7E to IRQ0E):** These bits select whether $IRQ_7$ to $IRQ_0$ are enabled or disabled.

| Bit n | |
|---|---|
| **IRQnE** | **Description** |
| 0 | $IRQ_n$ interrupts disabled (Initial value) |
| 1 | $IRQ_n$ interrupts enabled |

(n = 7 to 0)

### 5.2.5 IRQ Sense Control Registers H and L (ISCRH, ISCRL)

**ISCRH**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**ISCRL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The ISCR registers are 16-bit readable/writable registers that select rising edge, falling edge, or both edge detection, or level sensing, for the input at pins $\overline{IRQ_7}$ to $\overline{IRQ_0}$.

The ISCR registers are initialized to H'0000 by a reset and in hardware standby mode.

**Bits 15 to 0:** $IRQ_7$ Sense Control A and B (IRQ7SCA, IRQ7SCB) to $IRQ_0$ Sense Control A and B (IRQ0SCA, IRQ0SCB)

**Bits 15 to 0**

| IRQ7SCB to IRQ0SCB | IRQ7SCA to IRQ0SCA | Description |
|---|---|---|
| 0 | 0 | Interrupt request generated at $\overline{IRQ}_7$ to $\overline{IRQ}_0$ input low level (initial value) |
| | 1 | Interrupt request generated at falling edge of $\overline{IRQ}_7$ to $\overline{IRQ}_0$ input |
| 1 | 0 | Interrupt request generated at rising edge of $\overline{IRQ}_7$ to $\overline{IRQ}_0$ input |
| | 1 | Interrupt request generated at both falling and rising edges of $\overline{IRQ}_7$ to $\overline{IRQ}_0$ input |

### 5.2.6 IRQ Status Register (ISR)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Only 0 can be written, to clear the flag.

ISR is an 8-bit readable/writable register that indicates the status of $IRQ_7$ to $IRQ_0$ interrupt requests.

ISR is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—$IRQ_7$ to $IRQ_0$ flags (IRQ7F to IRQ0F):** These bits indicate the status of $IRQ_7$ to $IRQ_0$ interrupt requests.

| Bit n | |
|---|---|
| **IRQnF** | **Description** |
| 0 | [Clearing conditions] (Initial value)<br>• Cleared by reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag<br>• When interrupt exception handling is executed when low-level detection is set (IRQnSCB = IRQnSCA = 0) and $\overline{IRQ}_n$ input is high<br>• When IRQn interrupt exception handling is executed when falling, rising, or both-edge detection is set (IRQnSCB = 1 or IRQnSCA = 1)<br>• When the DTC is activated by an $IRQ_n$ interrupt, and the DISEL bit in MRB of the DTC is cleared to 0 |
| 1 | [Setting conditions]<br>• When $\overline{IRQ}_n$ input goes low when low-level detection is set (IRQnSCB = IRQnSCA = 0)<br>• When a falling edge occurs in $\overline{IRQ}_n$ input when falling edge detection is set (IRQnSCB = 0, IRQnSCA = 1)<br>• When a rising edge occurs in $\overline{IRQ}_n$ input when rising edge detection is set (IRQnSCB = 1, IRQnSCA = 0)<br>• When a falling or rising edge occurs in $\overline{IRQ}_n$ input when both-edge detection is set (IRQnSCB = IRQnSCA = 1) |

(n = 7 to 0)

## 5.3 Interrupt Sources

Interrupt sources comprise external interrupts (NMI and $IRQ_7$ to $IRQ_0$) and internal interrupts (52 sources).

### 5.3.1 External Interrupts

There are nine external interrupts: NMI and $IRQ_7$ to $IRQ_0$. Of these, NMI and $IRQ_2$ to $IRQ_0$ can be used to restore the H8S/2655 Series from software standby mode.

**NMI Interrupt:** NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

The vector number for NMI interrupt exception handling is 7.

**$IRQ_7$ to $IRQ_0$ Interrupts:** Interrupts $IRQ_7$ to $IRQ_0$ are requested by an input signal at pins $\overline{IRQ}_7$ to $\overline{IRQ}_0$. Interrupts $IRQ_7$ to $IRQ_0$ have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins IRQ7 to IRQ0.
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- The interrupt control level can be set with ICR, and the interrupt priority level can be set with IPR.
- The status of interrupt requests $IRQ_7$ to $IRQ_0$ is indicated in ISR. ISR flags can be cleared to 0 by software.

A block diagram of interrupts $IRQ_7$ to $IRQ_0$ is shown in figure 5-2.

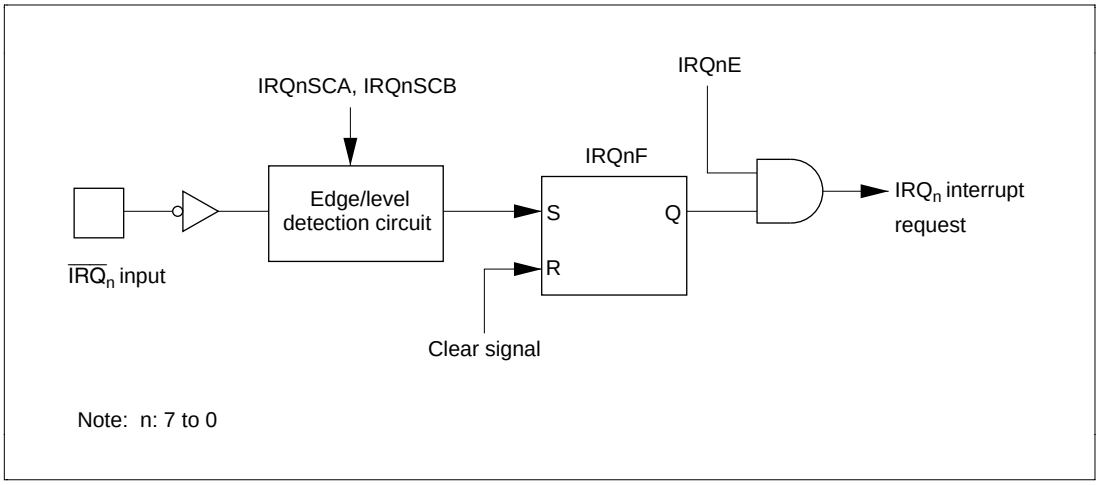



**Figure 5-2 Block Diagram of Interrupts $IRQ_7$ to $IRQ_0$**

Figure 5-3 shows the timing of setting IRQnF.

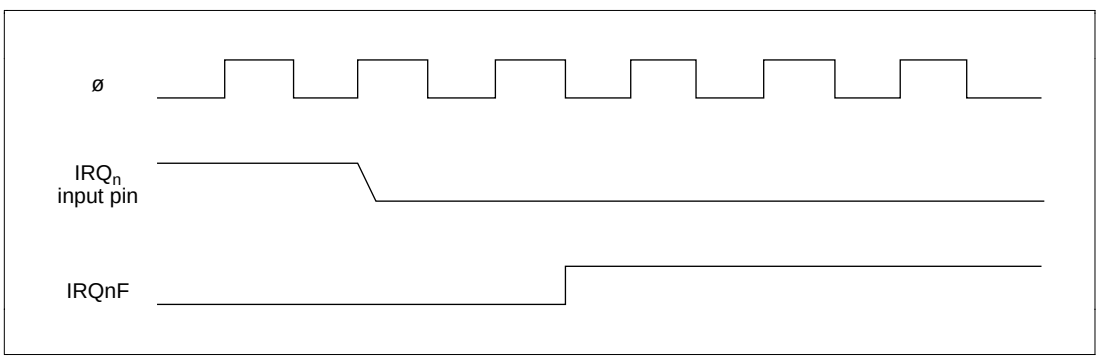



**Figure 5-3 Timing of Setting IRQnF**

The vector numbers for $IRQ_7$ to $IRQ_0$ interrupt exception handling are 23 to 16.

Detection of $IRQ_7$ to $IRQ_0$ interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding DDR to 0 and use the pin as an I/O pin for another function.

### 5.3.2 Internal Interrupts

There are 52 sources for internal interrupts from on-chip supporting modules.

- For each on-chip supporting module there are flags that indicate the interrupt request status, and enable bits that select enabling or disabling of these interrupts. If any one of these is set to 1, an interrupt request is issued to the interrupt controller.
- The interrupt control level can be set by means of ICR, and the interrupt priority level can be set by means of IPR.
- The DMAC and DTC can be activated by a TPU, 8-bit timer, SCI, or other interrupt request. When the DMAC or DTC is activated by an interrupt, the interrupt control mode and interrupt mask bits are not affected.

### 5.3.3 Interrupt Exception Handling Vector Table

Table 5-5 shows interrupt exception handling sources, vector addresses, and interrupt priorities. For default priorities, the lower the vector number, the higher the priority.

Priorities among modules can be set by means of the ICR and IPR. The situation when two or more modules are set to the same priority, and priorities within a module, are fixed as shown in table 5-5.

**Table 5-5 Interrupt Sources, Vector Addresses, and Interrupt Priorities**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address* Normal Mode | Vector Address* Advanced Mode | ICR | IPR | Priority |
|---|---|---|---|---|---|---|---|
| NMI | External pin | 7 | H'000E | H'001C | | | High |
| $IRQ_0$ | | 16 | H'0020 | H'0040 | ICRA7 | IPRA 6 to 4 | |
| $IRQ_1$ | | 17 | H'0022 | H'0044 | ICRA6 | IPRA 2 to 0 | |
| $IRQ_2$ | | 18 | H'0024 | H'0048 | ICRA5 | IPRB 6 to 4 | |
| $IRQ_3$ | | 19 | H'0026 | H'004C | | | |
| $IRQ_4$ | | 20 | H'0028 | H'0050 | ICRA4 | IPRB 2 to 0 | |
| $IRQ_5$ | | 21 | H'002A | H'0054 | | | |
| $IRQ_6$ | | 22 | H'002C | H'0058 | ICRA3 | IPRC 6 to 4 | |
| $IRQ_7$ | | 23 | H'002E | H'005C | | | |
| SWDTEND (software activation interrupt end) | DTC | 24 | H'0030 | H'0060 | ICRA2 | IPRC 2 to 0 | |
| WOVI (interval timer) | Watchdog timer | 25 | H'0032 | H'0064 | ICRA1 | IPRD 6 to 4 | |
| CMI (compare match) | Refresh controller | 26 | H'0034 | H'0068 | ICRA0 | IPRD 2 to 0 | |
| Reserved | — | 27 | H'0036 | H'006C | ICRB7 | IPRE 6 to 4 | |
| ADI (A/D conversion end) | A/D | 28 | H'0038 | H'0070 | ICRB6 | IPRE 2 to 0 | |
| Reserved | — | 29 | H'003A | H'0074 | | | |
| | | 30 | H'003C | H'0078 | | | |
| | | 31 | H'003E | H'007C | | | |
| TGI0A (TGR0A input capture/compare match) | TPU channel 0 | 32 | H'0040 | H'0080 | ICRB5 | IPRF 6 to 4 | |
| TGI0B (TGR0B input capture/compare match) | | 33 | H'0042 | H'0084 | | | |
| TGI0C (TGR0C input capture/compare match) | | 34 | H'0044 | H'0088 | | | |
| TGI0D (TGR0D input capture/compare match) | | 35 | H'0046 | H'008C | | | |
| TCI0V (overflow 0) | | 36 | H'0048 | H'0090 | | | |
| Reserved | — | 37 | H'004A | H'0094 | | | |
| | | 38 | H'004C | H'0098 | | | |
| | | 39 | H'004E | H'009C | | | Low |

Note: * Lower 16 bits of the start address.

**Table 5-5 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address* Normal Mode | Vector Address* Advanced Mode | ICR | IPR | Priority |
|---|---|---|---|---|---|---|---|
| TGI1A (TGR1A input capture/compare match) | TPU channel 1 | 40 | H'0050 | H'00A0 | ICRB4 | IPRF 2 to 0 | High |
| TGI1B (TGR1B input capture/compare match) | | 41 | H'0052 | H'00A4 | | | |
| TCI1V (overflow 1) | | 42 | H'0054 | H'00A8 | | | |
| TCI1U (underflow 1) | | 43 | H'0056 | H'00AC | | | |
| TGI2A (TGR2A input capture/compare match) | TPU channel 2 | 44 | H'0058 | H'00B0 | ICRB3 | IPRG 6 to 4 | |
| TGI2B (TGR2B input capture/compare match) | | 45 | H'005A | H'00B4 | | | |
| TCI2V (overflow 2) | | 46 | H'005C | H'00B8 | | | |
| TCI2U (underflow 2) | | 47 | H'005E | H'00BC | | | |
| TGI3A (TGR3A input capture/compare match) | TPU channel 3 | 48 | H'0060 | H'00C0 | ICRB2 | IPRG 2 to 0 | |
| TGI3B (TGR3B input capture/compare match) | | 49 | H'0062 | H'00C4 | | | |
| TGI3C (TGR3C input capture/compare match) | | 50 | H'0064 | H'00C8 | | | |
| TGI3D (TGR3D input capture/compare match) | | 51 | H'0066 | H'00CC | | | |
| TCI3V (overflow 1) | | 52 | H'0068 | H'00D0 | | | |
| Reserved | — | 53 | H'006A | H'00D4 | | | |
| | | 54 | H'006C | H'00D8 | | | |
| | | 55 | H'006E | H'00DC | | | |
| TGI4A (TGR4A input capture/compare match) | TPU channel 4 | 56 | H'0070 | H'00E0 | ICRB1 | IPRH 6 to 4 | |
| TGI4B (TGR4B input capture/compare match) | | 57 | H'0072 | H'00E4 | | | |
| TCI4V (overflow 4) | | 58 | H'0074 | H'00E8 | | | |
| TCI4U (underflow 4) | | 59 | H'0076 | H'00EC | | | |
| TGI5A (TGR5A input capture/compare match) | TPU channel 5 | 60 | H'0078 | H'00F0 | ICRB0 | IPRH 2 to 0 | |
| TGI5B (TGR5B input capture/compare match) | | 61 | H'007A | H'00F4 | | | |
| TCI5V (overflow 5) | | 62 | H'007C | H'00F8 | | | |
| TCI5U (underflow 5) | | 63 | H'007E | H'00FC | | | Low |

Note: * Lower 16 bits of the start address.

**Table 5-5 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address* Normal Mode | Vector Address* Advanced Mode | ICR | IPR | Priority |
|---|---|---|---|---|---|---|---|
| CMIA0 (compare match A0) | 8-bit timer | 64 | H'0080 | H'0100 | ICRC7 | IPRI | High |
| CMIB0 (compare match B0) | channel 0 | 65 | H'0082 | H'0104 | | 6 to 4 | ▲ |
| OVI0 (overflow 0) | | 66 | H'0084 | H'0108 | | | |
| Reserved | — | 67 | H'0086 | H'010C | | | |
| CMIA1 (compare match A1) | 8-bit timer | 68 | H'0088 | H'0110 | ICRC6 | IPRI | |
| CMIB1 (compare match B1) | channel 1 | 69 | H'008A | H'0114 | | 2 to 0 | |
| OVI1 (overflow 1) | | 70 | H'008C | H'0118 | | | |
| Reserved | — | 71 | H'008E | H'011C | | | |
| DEND0A (channel 0/ channel 0A transfer end) | DMAC | 72 | H'0090 | H'0120 | ICRC5 | IPRJ 6 to 4 | |
| DEND0B (channel 0B transfer end) | | 73 | H'0092 | H'0124 | | | |
| DEND1A (channel 1/ channel 1A transfer end) | | 74 | H'0094 | H'0128 | | | |
| DEND1B (channel 1B transfer end) | | 75 | H'0096 | H'012C | | | |
| Reserved | — | 76 | H'0098 | H'0130 | | | |
| | | 77 | H'009A | H'0134 | | | |
| | | 78 | H'009C | H'0138 | | | |
| | | 79 | H'009E | H'013C | | | |
| ERI0 (receive error 0) | SCI | 80 | H'00A0 | H'0140 | ICRC4 | IPRJ | |
| RXI0 (reception completed 0) | channel 0 | 81 | H'00A2 | H'0144 | | 2 to 0 | |
| TXI0 (transmit data empty 0) | | 82 | H'00A4 | H'0148 | | | |
| TEI0 (transmission end 0) | | 83 | H'00A6 | H'014C | | | |
| ERI1 (receive error 1) | SCI | 84 | H'00A8 | H'0150 | ICRC3 | IPRK | |
| RXI1 (reception completed 1) | channel 1 | 85 | H'00AA | H'0154 | | 6 to 4 | |
| TXI1 (transmit data empty 1) | | 86 | H'00AC | H'0158 | | | |
| TEI1 (transmission end 1) | | 87 | H'00AE | H'015C | | | |
| ERI2 (receive error 2) | SCI | 88 | H'00B0 | H'0160 | ICRC2 | IPRK | |
| RXI2 (reception completed 2) | channel 2 | 89 | H'00B2 | H'0164 | | 2 to 0 | |
| TXI2 (transmit data empty 2) | | 90 | H'00B4 | H'0168 | | | |
| TEI2 (transmission end 2) | | 91 | H'00B6 | H'016C | | | Low |

Note: * Lower 16 bits of the start address.

## 5.4 Interrupt Operation

### 5.4.1 Interrupt Control Modes and Interrupt Operation

Interrupt operations in the H8S/2655 Series differ depending on the interrupt control mode.

NMI interrupts are accepted at all times except in the reset state and the hardware standby state. In the case of IRQ interrupts and on-chip supporting module interrupts, an enable bit is provided for each interrupt. Clearing an enable bit to 0 disables the corresponding interrupt request. Interrupt sources for which the enable bits are set to 1 are controlled by the interrupt controller.

Table 5-6 shows the interrupt control modes.

The interrupt controller performs interrupt control according to the interrupt control mode set by the INTM1 and INTM0 bits in SYSCR, the priorities set in ICR and IPR, and the masking state indicated by the I and UI bits in the CPU's CCR, and bits I2 to I0 in EXR.

**Table 5-6 Interrupt Control Modes**

| Interrupt Control Mode | SYSCR | | Priority Setting Registers | Interrupt Mask Bits | Description |
|---|---|---|---|---|---|
| | INTM1 | INTM0 | | | |
| 0 | 0 | 0 | ICR | I | Interrupt mask control is performed by the I bit. Priority can be set with ICR. |
| 1 | | 1 | ICR | I, UI | 3-level interrupt mask control is performed by the I and UI bits. Priority can be set with ICR. |
| 2 | 1 | 0 | IPR | I2 to I0 | 8-level interrupt mask control is performed by bits I2 to I0. 8 priority levels can be set with IPR. |
| 3 | | 1 | ICR, IPR | I, UI, I2 to I0 | Control is performed by a combination of interrupt masking set by the I and UI bits and priority setting by ICR, based on 8-level interrupt mask control performed by bits I2 to I0 and 8-level priority setting by IPR. |

Figure 5-4 shows a block diagram of the priority decision circuit.

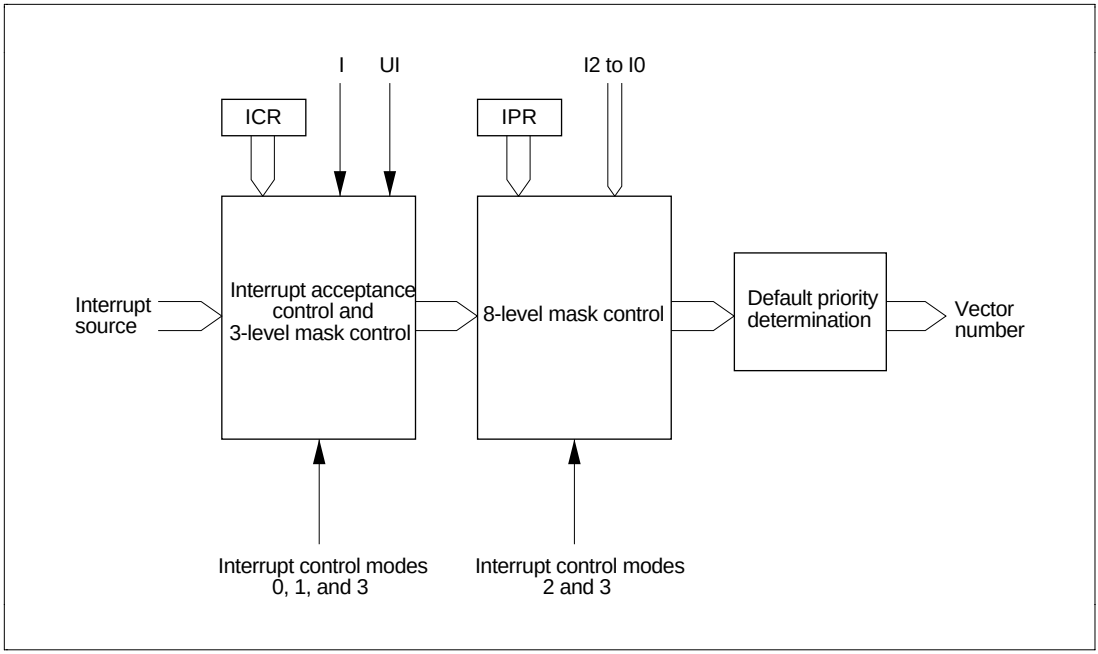



**Figure 5-4 Block Diagram of Interrupt Control Operation**

### (1) Interrupt Acceptance Control and 3-Level Control

In interrupt control modes 0, 1, and 3, interrupt acceptance control and 3-level mask control is performed by means of the I and UI bits in CCR, and ICR (control level).

Table 5-7 shows the interrupts selected in each interrupt control mode.

**Table 5-7 Interrupts Selected in Each Interrupt Control Mode (1)**

| | Interrupt Mask Bits | | |
|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **Selected Interrupts** |
| 0 | 0 | * | All interrupts (control level 1 has priority) |
| | 1 | * | NMI interrupts |
| 1 | 0 | * | All interrupts (control level 1 has priority) |
| | 1 | 0 | NMI and control level 1 interrupts |
| | | 1 | NMI interrupts |
| 2 | * | * | All interrupts |
| 3 | 0 | * | All interrupts |
| | 1 | 0 | NMI and control level 1 interrupts |
| | | 1 | NMI interrupts |

**Legend**

* : Don't care

### (2) 8-Level Control

In interrupt control modes 2 and 3, 8-level mask level determination is performed according to the interrupt priority level (IPR) for interrupts selected in interrupt acceptance control and 3-level control.

The interrupt source selected is the interrupt with the highest priority level, and whose priority level set in IPR is higher than the mask level.

**Table 5-8 Interrupts Selected in Each Interrupt Control Mode (2)**

| **Interrupt Control Mode** | **Selected Interrupts** |
|---|---|
| 0 | All interrupts |
| 1 | |
| 2 | Highest-priority-level (IPR) interrupt whose priority level is greater than the mask level (IPR > I2 to I0). |
| 3 | |

### (3) Default Priority Determination

When an interrupt is selected by 8-level control, its priority is determined and a vector number is generated.

If the same value is set for IPR and ICR, acceptance of multiple interrupts is enabled, and so only the interrupt source with the highest priority according to the preset default priorities is selected and has a vector number generated.

Interrupt sources with a lower priority than the accepted interrupt source are held pending.

Table 5-9 shows operations and control signal functions in each interrupt control mode.

**Table 5-9 Operations and Control Signal Functions in Each Interrupt Control Mode**

| Interrupt Control | Setting | | Interrupt Acceptance Control 3-Level Control | | | | 8-Level Control | | Default Priority | T |
|---|---|---|---|---|---|---|---|---|---|---|
| Mode | INTM1 | INTM0 | I | UI | ICR | | I2 to I0 | IPR | Determination | (Trace) |
| 0 | 0 | 0 | ◯ IM | — | PR | X | — | —*2 | ◯ | — |
| 1 | | 1 | ◯ IM | IM | PR | X | — | —*2 | ◯ | — |
| 2 | 1 | 0 | X —*1 | — | — | ◯ | IM | PR | ◯ | T |
| 3 | | 1 | ◯ IM | IM | PR | ◯ | IM | PR | ◯ | T |

**Legend**

◯ : Interrupt operation control performed
X : No operation. (All interrupts enabled)
IM : Used as interrupt mask bit
PR : Sets priority.
— : Not used.
*1 : Set to 1 when interrupt is accepted.
*2 : Keep the initial setting.

### 5.4.2 Interrupt Control Mode 0

Enabling and disabling of IRQ interrupts and on-chip supporting module interrupts can be set by means of the I bit in the CPU's CCR, and ICR. Interrupts are enabled when the I bit is cleared to 0, and disabled when set to 1. Control level 1 interrupt sources have higher priority.

Figure 5-5 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] When interrupt requests are sent to the interrupt controller, a control level 1 interrupt, according to the control level set in ICR, has priority for selection, and other interrupt requests are held pending. If a number of interrupt requests with the same control level setting are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 5-5 is selected.

[3] The I bit is then referenced. If the I bit is cleared to 0, the interrupt request is accepted. If the I bit is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending.

[4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[5] The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[6] Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.

[7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

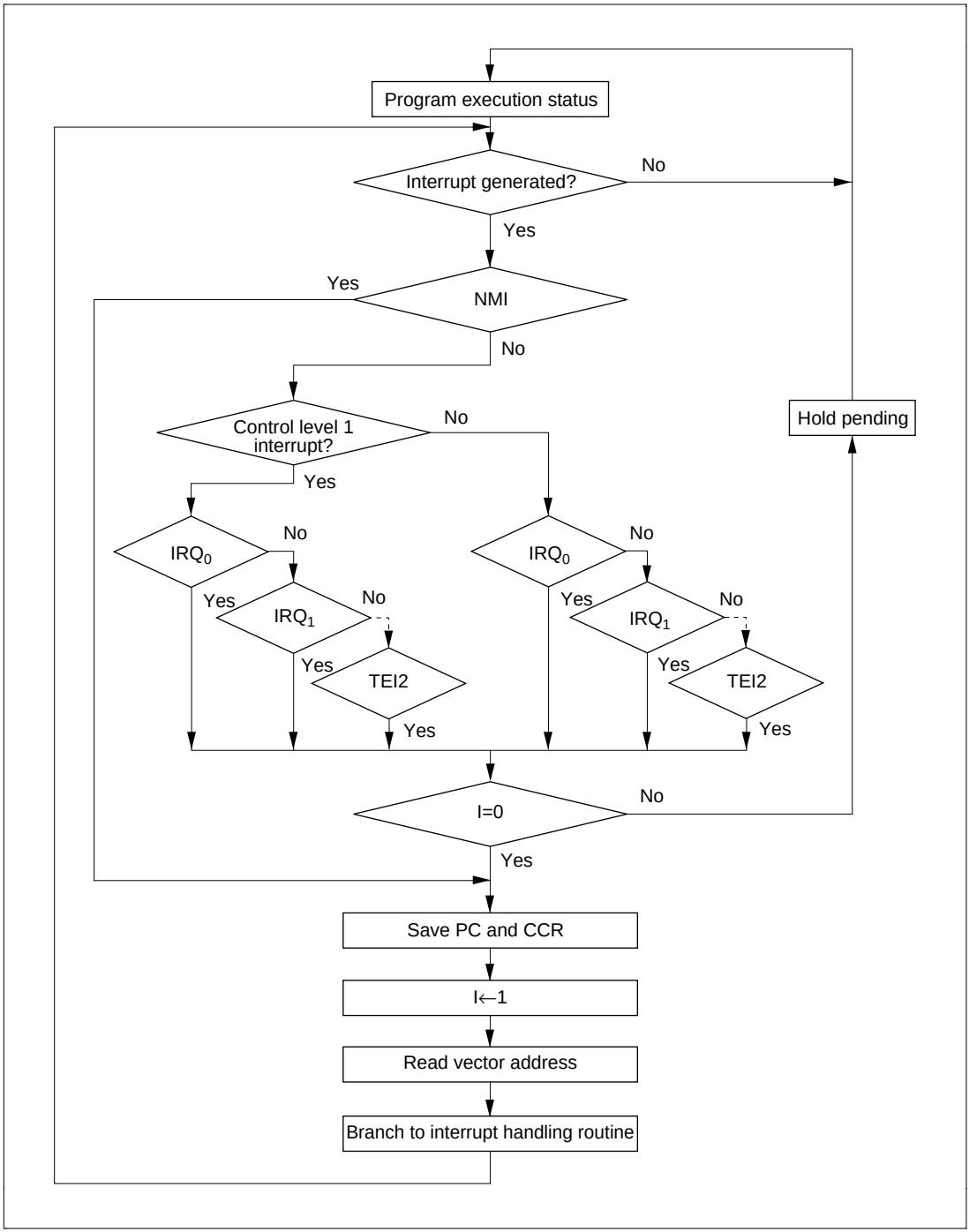



**Figure 5-5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

### 5.4.3 Interrupt Control Mode 1

Three-level masking is implemented for IRQ interrupts and on-chip supporting module interrupts by means of the I and UI bits in the CPU's CCR, and ICR.

- Control level 0 interrupt requests are enabled when the I bit is cleared to 0, and disabled when set to 1.
- Control level 1 interrupt requests are enabled when the I bit or UI bit is cleared to 0, and disabled when both the I bit and the UI bit are set to 1.

For example, if the interrupt enable bit for an interrupt request is set to 1, and H'20, H'00, and H'00 are set in ICRA, ICRB, and ICRC, respectively, (i.e. $IRQ_2$ and $IRQ_3$ interrupts are set to control level 1 and other interrupts to control level 0), the situation is as follows:

- When I = 0, all interrupts are enabled
  (Priority order: NMI > $IRQ_2$ > $IRQ_3$ > $IRQ_0$ ...)
- When I = 1 and UI = 0, only NMI, IRQ2, and IRQ3 interrupts are enabled
- When I = 1 and UI = 1, only NMI interrupts are enabled

Figure 5-6 shows the state transitions in these cases.

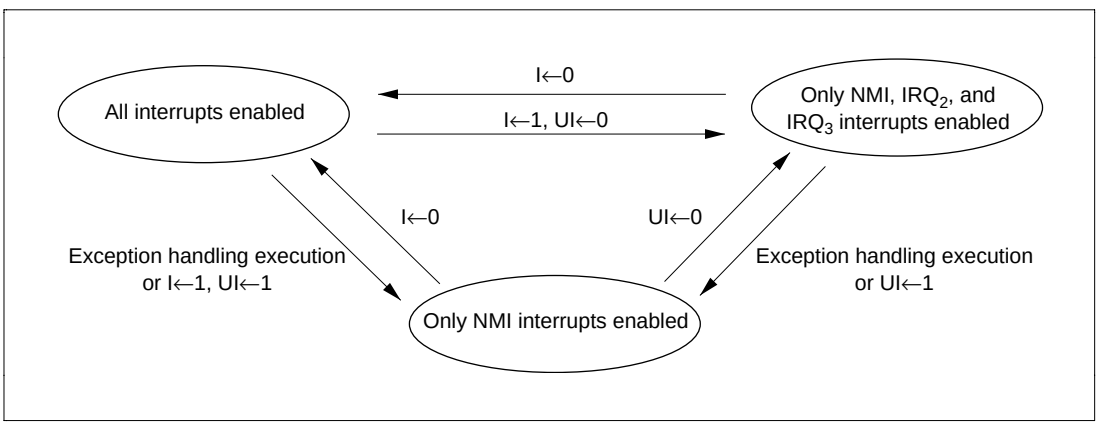



**Figure 5-6 Example of State Transitions in Interrupt Control Mode 1**

Figure 5-7 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] When interrupt requests are sent to the interrupt controller, a control level 1 interrupt, according to the control level set in ICR, has priority for selection, and other interrupt requests are held pending. If a number of interrupt requests with the same control level setting are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 5-5 is selected.

[3] The I bit is then referenced. If the I bit is cleared to 0, the UI bit is not affected.

An interrupt request set to interrupt control level 0 is accepted when the I bit is cleared to 0. If the I bit is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending.

An interrupt request set to interrupt control level 1 has priority over an interrupt request set to interrupt control level 0, and is accepted if the I bit is cleared to 0, or if the I bits is set to 1 and the UI bit is cleared to 0.

When both the I bit and the UI bit are set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending.

[4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[5] The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[6] Next, the I and UI bits in CCR are set to 1. This masks all interrupts except NMI.

[7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

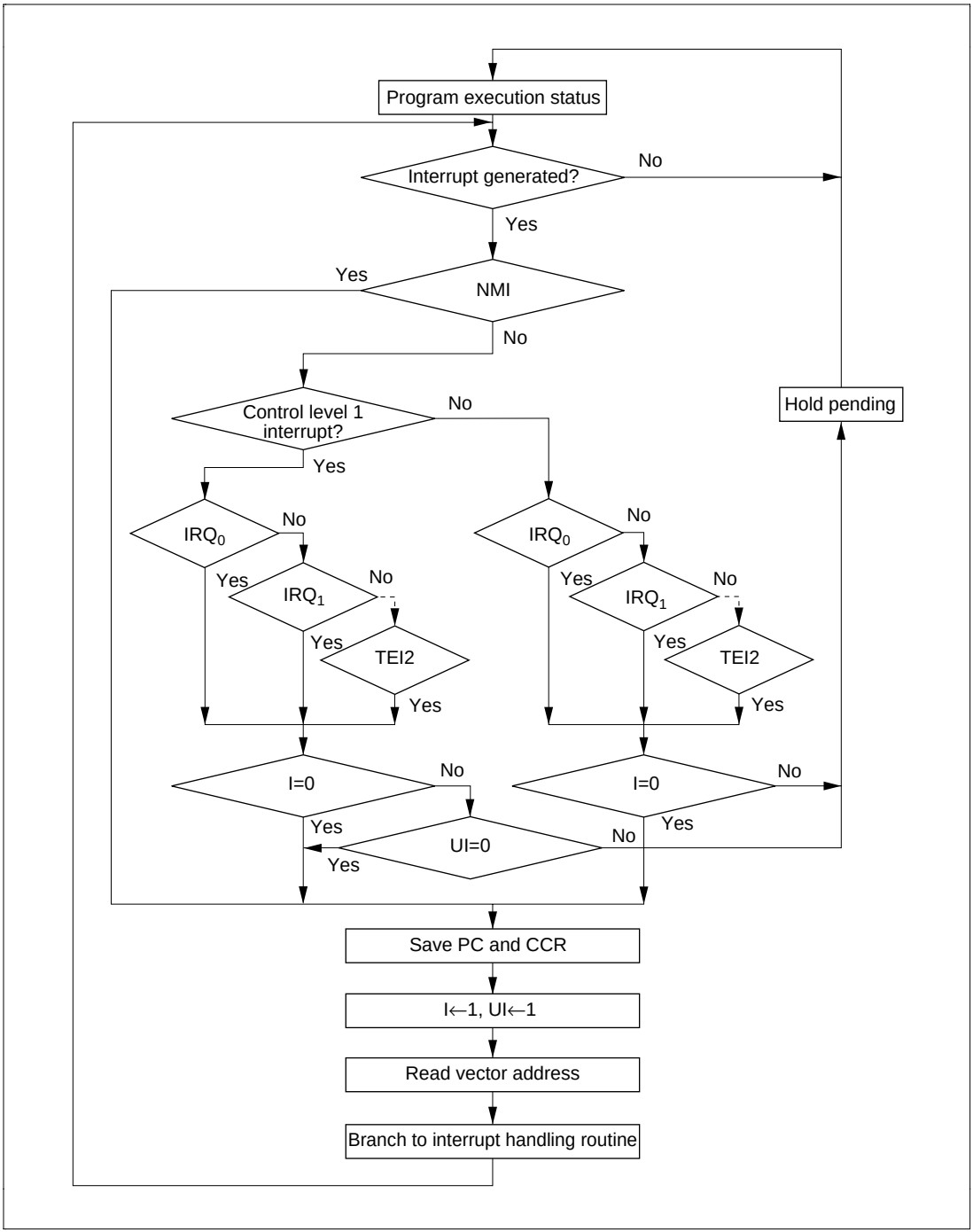



**Figure 5-7 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1**

### 5.4.4 Interrupt Control Mode 2

Eight-level masking is implemented for IRQ interrupts and on-chip supporting module interrupts by comparing the interrupt mask level set by bits I2 to I0 of EXR in the CPU with IPR.

Figure 5-8 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] When interrupt requests are sent to the interrupt controller, the interrupt with the highest priority according to the interrupt priority levels set in IPR is selected, and lower-priority interrupt requests are held pending. If a number of interrupt requests with the same priority are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 5-5 is selected.

[3] Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. An interrupt request with a priority no higher than the mask level set at that time is held pending, and only an interrupt request with a priority higher than the interrupt mask level is accepted.

[4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[5] The PC, CCR, and EXR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[6] The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority level of the accepted interrupt.

If the accepted interrupt is NMI, the interrupt mask level is set to H'7.

[7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

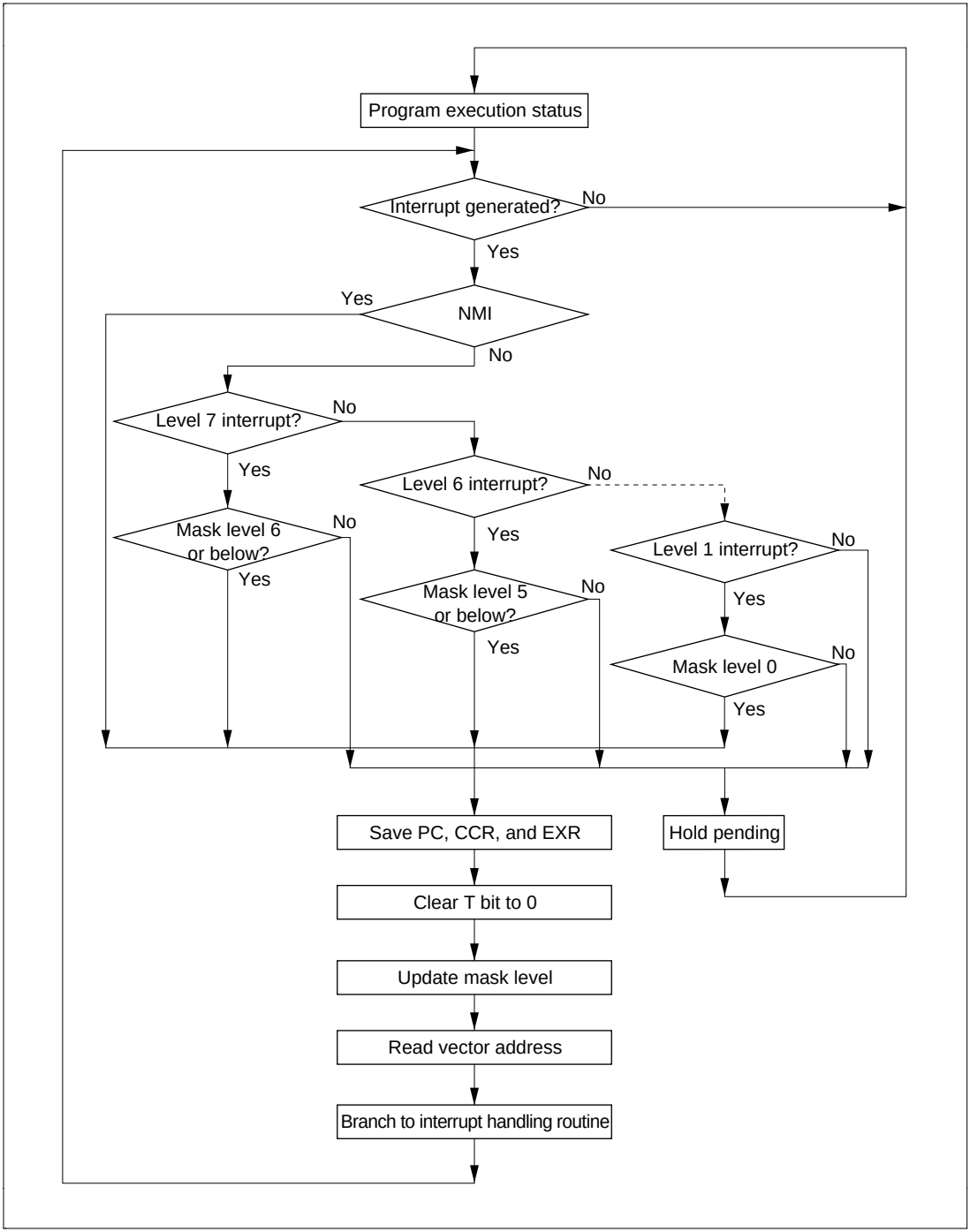



**Figure 5-8 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**

### 5.4.5 Interrupt Control Mode 3

Control of IRQ interrupts and on-chip supporting module interrupts is performed by a combination of interrupt masking set by the I and UI bits and control level setting by ICR, based on 8-level interrupt mask control performed by comparing the interrupt mask level in the CPU's EXR (bits I2 to I0) and the priority set in IPR.

- Control level 0 interrupt requests are enabled when the I bit is cleared to 0, and disabled when set to 1.
- Control level 1 interrupt requests are enabled when the I bit or UI bit is cleared to 0, and disabled when both the I bit and the UI bit are set to 1.
- Eight-level priority control is performed when the I bit is cleared to 0.

For example, if the interrupt enable bit for an interrupt request is set to 1, and H'00, H'30, and H'10 are set in ICRA, ICRB, and ICRC, respectively, (i.e. TPU channels 0 and 1 and SCI channel 0 are set to control level 1 and other interrupts to control level 0), the situation is as follows:

- When I = 0, 8-level mask control is performed for all interrupts.
- The interrupt controller enables TPU0, TPU1, and SCI0 interrupts. Bits I2 to I0 are disabled, and the interrupt mask level is regarded as 0.
- When I = 1 and UI = 1, only NMI interrupts are enabled.

Figure 5-9 shows the state transitions in these cases.

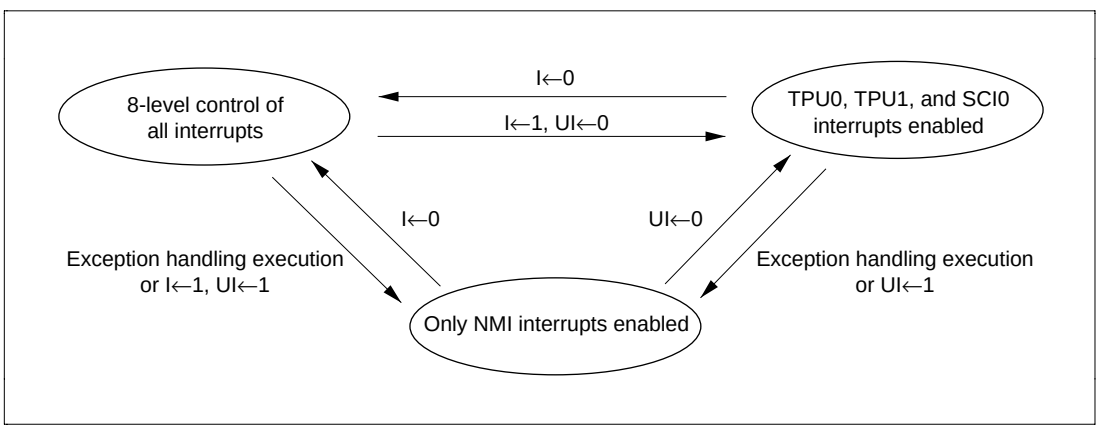



**Figure 5-9 Example of State Transitions in Interrupt Control Mode 3**

Figure 5-10 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] When interrupt requests are sent to the interrupt controller, it performs interrupt acceptance control.

If the I bit is cleared to 0, all interrupts are accepted.

If the I bit is set to 1 and the UI bit is cleared to 0, control level 1 interrupts are accepted.

If both the I bit and the UI bit are set to 1, only an NMI interrupt is accepted.

[3] The interrupt request with the highest priority according to the priority levels set in IPR is selected.

[4] If the I bit is cleared to 0, the priority level of the selected interrupt request is compared with the interrupt mask level set in bits I2 to I0. An interrupt request with a priority no higher than the mask level set at that time is held pending, and only an interrupt request with a priority higher than the interrupt mask level is accepted.

[5] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[6] The PC, CCR, and EXR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[7] Next, the I and UI bits in CCR are set to 1. This masks all interrupts except NMI. Also, bits I2 to I0 are rewritten with the priority of the accepted interrupt. If the accepted interrupt is NMI, the interrupt mask level is set to H'7.

[8] The T bit in EXR is cleared to 0.

[9] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

[10]If interrupts are enabled again in the interrupt handling routine, the control level of the interrupt to be enabled is set to 1, and the UI bit in CCR is cleared to 0. At control level 1, the interrupt with the highest priority according to the priority level is selected. Bits I2 to I0 are disabled, and the interrupt mask level is regarded as 0.

When the I bit is cleared to 0, the control level is ignored and an interrupt with a priority level higher than the mask level set in bits I2 to I0 is accepted.

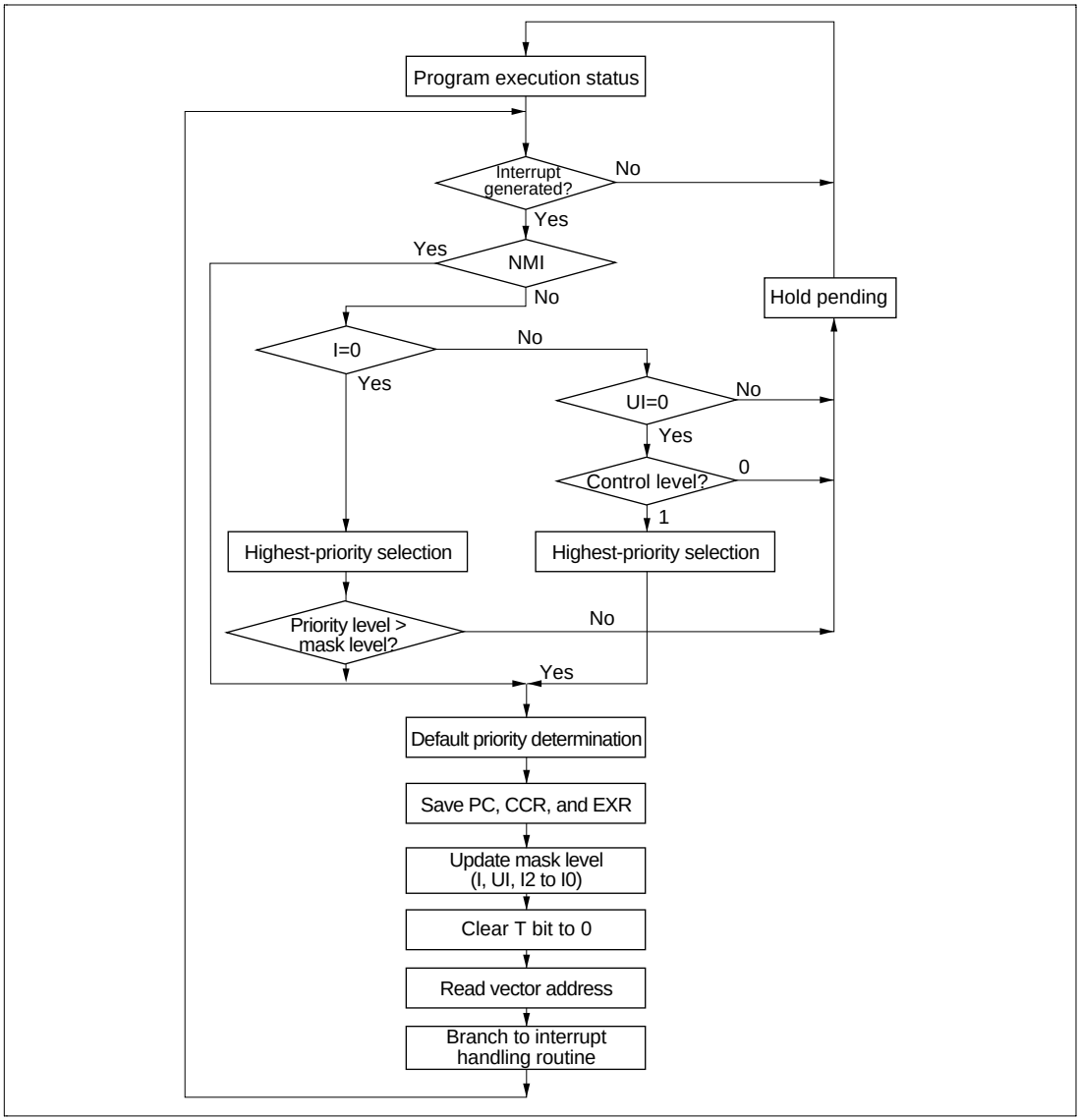
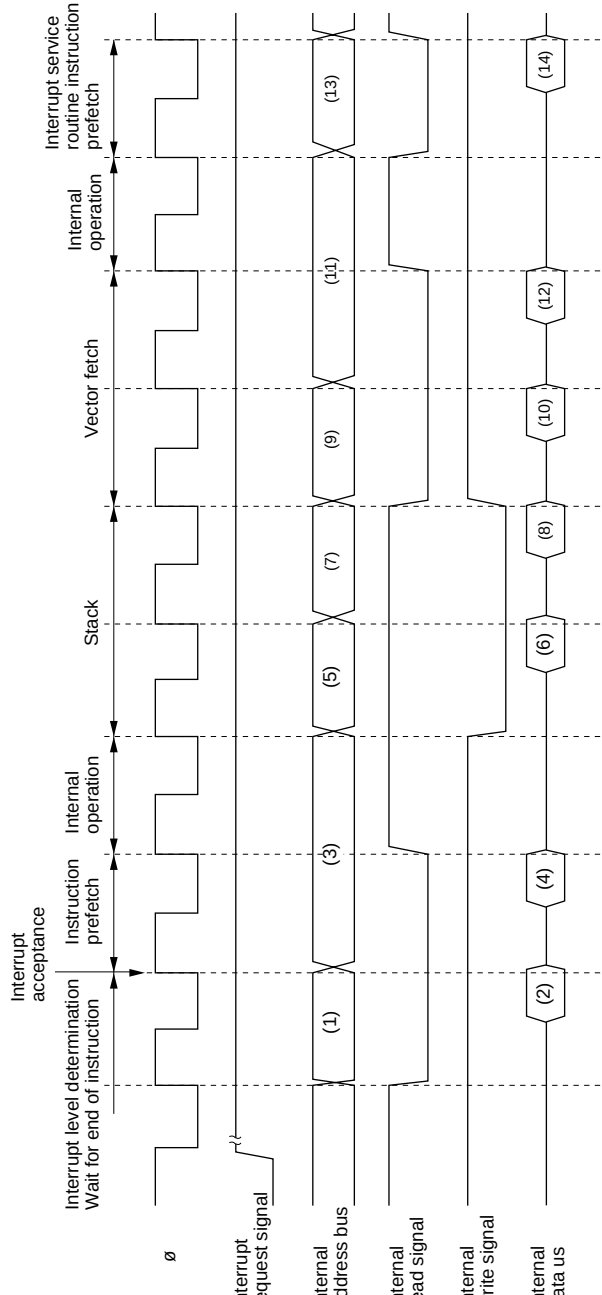



**Figure 5-10 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 3**

### 5.4.6 Interrupt Exception Handling Sequence

Figure 5-11 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.




(1) Instruction prefetch address (Not executed. This is the contents of the saved PC, the return address.)
(2) (4) Instruction code (Not executed.)
(3) Instruction prefetch address (Not executed.)
(5) SP-2
(7) SP-4
(6) (8) Saved PC and saved CCR
(9) (11) Vector address
(10) (12) Interrupt handling routine start address (vector address contents)
(13) Interrupt handling routine start address ((13) = (10) (12))
(14) First instruction of interrupt handling routine

**Figure 5-11 Interrupt Exception Handling**

### 5.4.7 Interrupt Response Times

The H8S/2655 Series is capable of fast word transfer instruction to on-chip memory, and the program area is provided in on-chip ROM and the stack area in on-chip RAM, enabling high-speed processing.

Table 5-10 shows interrupt response times - the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The execution status symbols used in table 5-11 are explained in table 5-11.

**Table 5-10 Interrupt Response Times**

| | | Normal Mode | | Advanced Mode | |
|---|---|---|---|---|---|
| **No.** | **Execution Status** | **INTM1 = 0** | **INTM1 = 1** | **INTM1 = 0** | **INTM1 = 1** |
| 1 | Interrupt priority determination*[1] | 3 | 3 | 3 | 3 |
| 2 | Number of wait states until executing instruction ends*[2] | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ |
| 3 | PC, CCR, EXR stack save | $2\cdot S_K$ | $3\cdot S_K$ | $2\cdot S_K$ | $3\cdot S_K$ |
| 4 | Vector fetch | $S_I$ | $S_I$ | $2\cdot S_I$ | $2\cdot S_I$ |
| 5 | Instruction fetch*[3] | $2\cdot S_I$ | $2\cdot S_I$ | $2\cdot S_I$ | $2\cdot S_I$ |
| 6 | Internal processing*[4] | 2 | 2 | 2 | 2 |
| Total (using on-chip memory) | | 11 to 31 | 12 to 32 | 12 to 32 | 13 to 33 |

Notes: 1. Two states in case of internal interrupt.
2. Refers to DIVXS instruction.
3. Prefetch after interrupt acceptance and interrupt handling routine prefetch.
4. Internal processing after interrupt acceptance and internal processing after vector fetch.

**Table 5-11 Number of States in Interrupt Handling Routine Execution Statuses**

| | | Object of Access | | | | |
|---|---|---|---|---|---|---|
| | | | External Device | | | |
| | | | 8 Bit Bus | | 16 Bit Bus | |
| **Symbol** | | **Internal Memory** | **2-State Access** | **3-State Access** | **2-State Access** | **3-State Access** |
| Instruction fetch | $S_I$ | 1 | 4 | 6+2m | 2 | 3+m |
| Branch address read | $S_J$ | | | | | |
| Stack manipulation | $S_K$ | | | | | |

**Legend**

m : Number of wait states in an external device access.

## 5.5 Usage Notes

### 5.5.1 Contention between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupts, the disabling becomes effective after execution of the instruction.

In other words, when an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored.

The same also applies when an interrupt source flag is cleared.

Figure 5-12 shows and example in which the CMIEA bit in 8-bit timer TCR is cleared to 0.

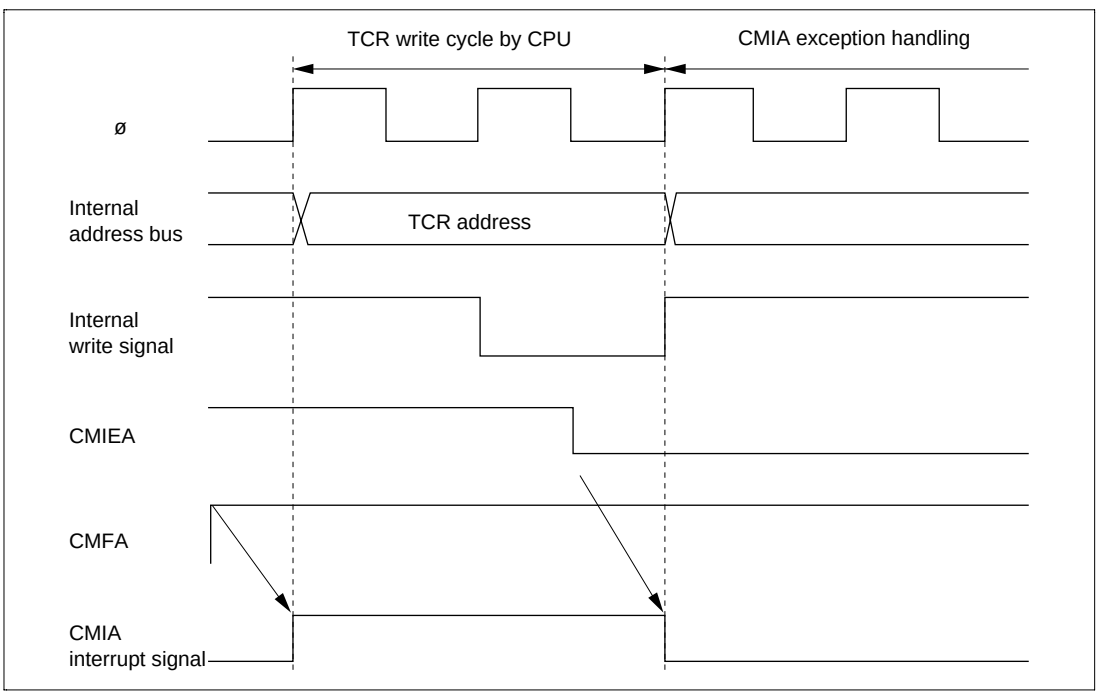



**Figure 5-12 Contention between Interrupt Generation and Disabling**

The above contention will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

### 5.5.2 Instructions that Disable Interrupts

Instructions that disable interrupts are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts including NMI are disabled and the next instruction is always executed. When the I bit or a UI bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

### 5.5.3 Times when Interrupts are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU has updated the mask level with an LDC, ANDC, ORC, or XORC instruction.

### 5.5.4 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B instruction and the EEPMOV.W instruction.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the move is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at a break in the transfer cycle. The PC value saved on the stack in this case is the address of the next instruction.

Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1:   EEPMOV.W
      MOV.W     R4,R4
      BNE       L1
```

## 5.6 DTC and DMAC Activation by Interrupt

### 5.6.1 Overview

The DTC and DMAC can be activated by an interrupt. In this case, the following options are available:

- Interrupt request to CPU
- Activation request to DTC
- Activation request to DMAC
- Selection of a number of the above

For details of interrupt requests that can be used with to activate the DTC or DMAC, see section 8, Data Transfer Controller, and section 7, DMA Controller.

### 5.6.2 Block Diagram

Figure 5-13 shows a block diagram of the DTC and DMAC interrupt controller.

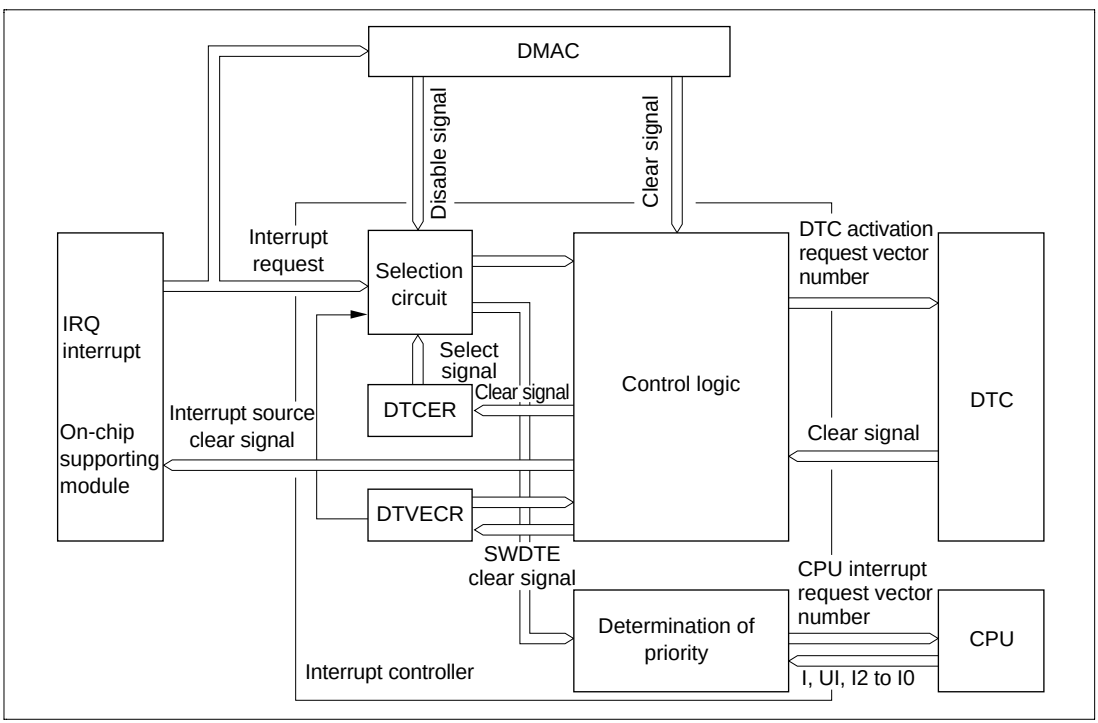



**Figure 5-13 Interrupt Control for DTC and DMAC**

### 5.6.3 Operation

The interrupt controller has three main functions in DTC and DMAC control.

**(1) Selection of Interrupt Source:** With the DMAC, the activation source is input directly to each channel. The activation source for each DMAC channel is selected with bits DTF3 to DTF0 in DMACR. Whether the selected activation source is to be managed by the DMAC can be selected with the DTA bit of DMABCR. When the DTA bit is set to 1, the interrupt source constituting that DMAC activation source is not a DTC activation source or CPU interrupt source.

For interrupt sources other than interrupts managed by the DMAC, it is possible to select DTC activation request or CPU interrupt request with the DTCE bit of DTCEA to DTCEF in the DTC.

After a DTC data transfer, the DTCE bit can be cleared to 0 and an interrupt request sent to the CPU in accordance with the specification of the DISEL bit of MRB in the DTC.

When the DTC has performed the specified number of data transfers and the transfer counter value is zero, the DTCE bit is cleared to 0 and an interrupt request is sent to the CPU after the DTC data transfer.

**(2) Determination of Priority:** The DTC activation source is selected in accordance with the default priority order, and is not affected by mask or priority levels. See section 7.6, Interrupts, and section 8.3.3, DTC Vector Table, for the respective priorities.

With the DMAC, the activation source is input directly to each channel.

**(3) Operation Order:** If the same interrupt is selected as a DTC activation source and a CPU interrupt source, the DTC data transfer is performed first, followed by CPU interrupt exception handling.

If the same interrupt is selected as a DMAC activation source and a DTC activation source or CPU interrupt source, operations are performed for them independently according to their respective operating statuses and bus mastership priorities.

Table 5-12 summarizes interrupt source selection and interrupt source clearance control according to the settings of the DTA bit of DMABCR in the DMAC, the DTCE bit of DTCEA to DTCEF in the DTC and the DISEL bit of MRB in the DTC.

**Table 5-12 Interrupt Source Selection and Clearing Control**

| Settings | | | | | |
|---|---|---|---|---|---|
| **DMAC** | **DTC** | **Interrupt Source Selection/Clearing Control** | | | |
| **DTA** | **DTCE** | **DISEL** | **DMAC** | **DTC** | **CPU** |
| 0 | 0 | * | ◯ | X | Δ |
| | 1 | 0 | ◯ | Δ | X |
| | | 1 | ◯ | ◯ | Δ |
| 1 | * | * | Δ | X | X |

**Legend**

Δ : The relevant interrupt is used. Interrupt source clearing is performed.
(The CPU should clear the source flag in the interrupt handling routine.)

◯ : The relevant interrupt is used. The interrupt source is not cleared.

X : The relevant bit cannot be used.

* : Don't care

**(4) Notes on Use:** SCI and A/D converter interrupt sources are cleared when the DMAC or DTC reads or writes to the prescribed register, and are not dependent upon the DTA bit or DISEL bit.

# Section 6 Bus Controller

## 6.1 Overview

The H8S/2655 Series has a built-in bus controller (BSC) that manages the external address space divided into eight areas. The bus specifications, such as bus width and number of access states, can be set independently for each area, enabling multiple memories to be connected easily.

The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters: the CPU, DMA controller (DMAC), and data transfer controller (DTC).

### 6.1.1 Features

The features of the bus controller are listed below.

- Manages external address space in area units
  - In advanced mode, manages the external space as 8 areas of 128-kbytes/2-Mbytes
  - In normal mode, manages the external space as a single area
  - Bus specifications can be set independently for each area
  - DRAM/PSRAM/burst ROM interfaces can be set
- Basic bus interface
  - Chip select ($\overline{CS}_0$ to $\overline{CS}_7$) can be output for areas 0 to 7
  - 8-bit access or 16-bit access can be selected for each area
  - 2-state access or 3-state access can be selected for each area
  - Program wait states can be inserted for each area
- DRAM interface
  - DRAM interface can be set for areas 2 to 5 (in advanced mode)
  - Row address/column address multiplexed output (8/9/10 bits)
  - Two byte access methods (2-CAS and 2-WE)
  - Burst operation (fast page mode)
  - $T_P$ cycle insertion to secure RAS precharging time
  - Choice of CAS-before-RAS refreshing or self-refreshing

- Pseudo-SRAM (PSRAM) direct interface
  - PSRAM interface can be set for areas 2 to 5 (in advanced mode)
  - Burst operation (static column mode)
  - $T_P$ cycle insertion to secure RAS precharging time
  - Choice of auto-refreshing or self-refreshing

- Burst ROM interface
  - Burst ROM interface can be set for area 0
  - Choice of 1- or 2-state burst access

- Idle cycle insertion
  - An idle cycle can be inserted in case of an external read cycle between different areas
  - An idle cycle can be inserted in case of an external write cycle immediately after an external read cycle

- Write buffer functions
  - External write cycle and internal access can be executed in parallel
  - DMAC single-address mode and internal access can be executed in parallel

- Bus arbitration function
  - Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, and DTC

- Other features
  - Refresh counter (refresh timer) can be used as an interval timer
  - External bus release function

### 6.1.2 Block Diagram

Figure 6-1 shows a block diagram of the bus controller.

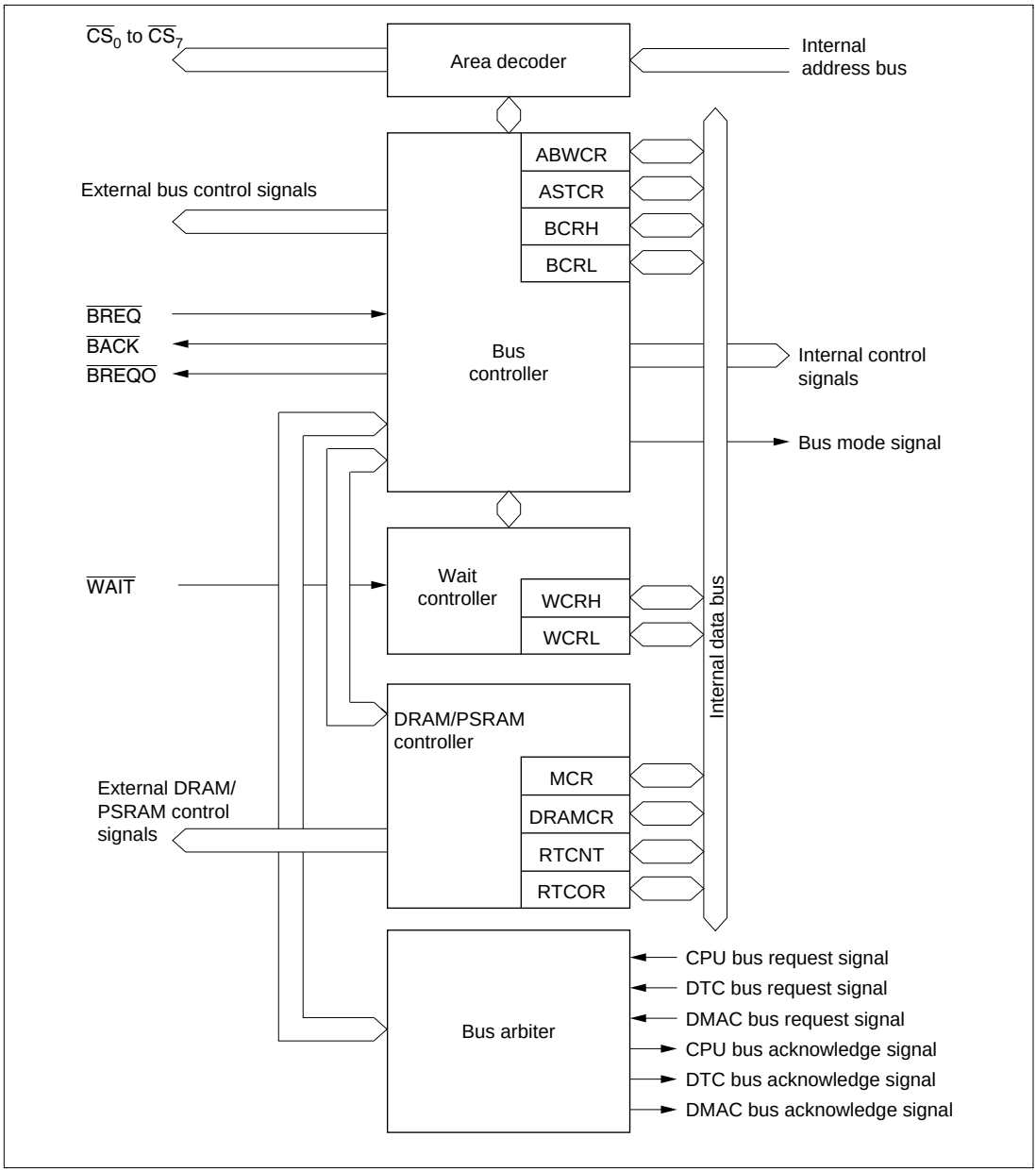



**Figure 6-1 Block Diagram of Bus Controller**

### 6.1.3 Pin Configuration

Table 6-1 summarizes the pins of the bus controller.

**Table 6-1 Bus Controller Pins**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Address strobe | $\overline{AS}$ | Output | Strobe signal indicating that address output on address bus is enabled. |
| Read | $\overline{RD}$ | Output | Strobe signal indicating that external space is being read. |
| High write/write enable/upper write enable | $\overline{HWR}$ | Output | Strobe signal indicating that external space is to be written, and upper half ($D_{15}$ to $D_8$) of data bus is enabled.<br>2-CAS DRAM write enable signal.<br>2-WE DRAM upper write enable signal. |
| Low write/lower column address strobe/lower write enable | $\overline{LWR}$ | Output | Strobe signal indicating that external space is to be written, and lower half ($D_7$ to $D_0$) of data bus is enabled.<br>2-CAS (LCASS = 1) DRAM lower column address strobe signal.*<br>2-WE DRAM lower write enable signal. |
| Chip select 0 | $\overline{CS}_0$ | Output | Strobe signal indicating that area 0 is selected. |
| Chip select 1 | $\overline{CS}_1$ | Output | Strobe signal indicating that area 1 is selected. |
| Chip select 2/row address strobe 2 | $\overline{CS}_2$ | Output | Strobe signal indicating that area 2 is selected.<br>DRAM row address strobe signal when area 2 is in DRAM space. |
| Chip select 3/row address strobe 3 | $\overline{CS}_3$ | Output | Strobe signal indicating that area 3 is selected.<br>DRAM row address strobe signal when area 3 is in DRAM space. |

Note: * Using the LCASS bit in BCRL, it is possible to select use of either the $\overline{LWR}$ pin or the $\overline{LCAS}$ pin for the 2-CAS type DRAM lower column strobe signal.

**Table 6-1 Bus Controller Pins (cont)**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Chip select 4/row address strobe 4 | $\overline{CS}_4$ | Output | Strobe signal indicating that area 4 is selected.<br>DRAM row address strobe signal when area 4 is in DRAM space. |
| Chip select 5/row address strobe 5 | $\overline{CS}_5$ | Output | Strobe signal indicating that area 5 is selected.<br>DRAM row address strobe signal when area 5 is in DRAM space. |
| Chip select 6 | $\overline{CS}_6$ | Output | Strobe signal indicating that area 6 is selected. |
| Chip select 7 | $\overline{CS}_7$ | Output | Strobe signal indicating that area 7 is selected. |
| Upper column address strobe/ column address strobe/output enable/refresh | $\overline{CAS}$/ $\overline{OE}$ | Output | 2-CAS DRAM upper column address strobe signal.<br>2-WE DRAM column address strobe signal.<br>PSRAM output enable signal when areas 2 to 5 are in PSRAM space. |
| Lower column strobe | $\overline{LCAS}$ | Output | The 2-CAS type (LCASS = 0) DRAM lower column address strobe signal.* |
| Wait | $\overline{WAIT}$ | Input | Wait request signal when accessing external 3-state access space. |
| Bus request | $\overline{BREQ}$ | Input | Request signal that releases bus to external device. |
| Bus request acknowledge | $\overline{BACK}$ | Output | Acknowledge signal indicating that bus has been released. |
| Bus request output | $\overline{BREQO}$ | Output | External bus request signal used when internal bus master accesses external space when external bus is released. |

Note: * Using the LCASS bit in BCRL, it is possible to select use of either the $\overline{LWR}$ pin or the $\overline{LCAS}$ pin for the 2-CAS type DRAM lower column strobe signal.

### 6.1.4 Register Configuration

Table 6-2 summarizes the registers of the bus controller.

**Table 6-2 Bus Controller Registers**

| Name | Abbreviation | R/W | Initial Value | | Address*[1] |
|---|---|---|---|---|---|
| | | | Power-On Reset | Manual Reset | |
| Bus width control register | ABWCR | R/W | H'FF/H'00*[2] | Retained | H'FED0 |
| Access state control register | ASTCR | R/W | H'FF | Retained | H'FED1 |
| Wait control register H | WCRH | R/W | H'FF | Retained | H'FED2 |
| Wait control register L | WCRL | R/W | H'FF | Retained | H'FED3 |
| Bus control register H | BCRH | R/W | H'D0 | Retained | H'FED4 |
| Bus control register L | BCRL | R/W | H'3C | Retained | H'FED5 |
| Memory control register | MCR | R/W | H'00 | Retained | H'FED6 |
| DRAM control register | DRAMCR | R/W | H'00 | Retained | H'FED7 |
| Refresh timer/counter | RTCNT | R/W | H'00 | Retained | H'FED8 |
| Refresh time constant register | RTCOR | R/W | H'FF | Retained | H'FED9 |

Notes: 1. Lower 16 bits of the address.
2. Determined by the MCU operating mode.

## 6.2 Register Descriptions

### 6.2.1 Bus Width Control Register (ABWCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 |
| Modes 1 to 3, 5 to 7 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RW | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Mode 4 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ABWCR is an 8-bit readable/writable register that designates each area for either 8-bit access or 16-bit access.

ABWCR sets the data bus width for the external memory space. The bus width for on-chip memory and internal I/O registers is fixed regardless of the settings in ABWCR.

In normal mode, the settings of bits ABW7 to ABW1 have no effect on operation.

After a power-on reset and in hardware standby mode, ABWCR is initialized to H'FF in modes 1, 2, 3, and 5, 6, 7, and to H'00 in mode 4. It is not initialized by a manual reset or in software standby mode.

**Bits 7 to 0—Area 7 to 0 Bus Width Control (ABW7 to ABW0):** These bits select whether the corresponding area is to be designated for 8-bit access or 16-bit access. In normal mode, only part of area 0 is enabled, and the ABW0 bit selects whether external space is to be designated for 8-bit access or 16-bit access .

**Bit n**

| ABWn | Description |
|---|---|
| 0 | Area n is designated for 16-bit access |
| 1 | Area n is designated for 8-bit access |

(n = 7 to 0)

### 6.2.2 Access State Control Register (ASTCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ASTCR is an 8-bit readable/writable register that designates each area as either a 2-state access space or a 3-state access space.

ASTCR sets the number of access states for the external memory space. The number of access states for on-chip memory and internal I/O registers is fixed regardless of the settings in ASTCR.

In normal mode, the settings of bits AST7 to AST1 have no effect on operation.

ASTCR is initialized to H'FF by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bits 7 to 0—Area 7 to 0 Access State Control (AST7 to AST0):** These bits select whether the corresponding area is to be designated as a 2-state access space or a 3-state access space. In normal mode, only part of area 0 is enabled, and the AST0 bit selects whether external space is to be designated for 2-state access or 3-state access.

Wait state insertion is enabled or disabled at the same time.

| **Bit n** | | |
|---|---|---|
| **ASTn** | **Description** | |
| 0 | Area n is designated for 2-state access<br>Wait state insertion in area n external space is disabled | |
| 1 | Area n is designated for 3-state access<br>Wait state insertion in area n external space is enabled | (Initial value) |

(n = 7 to 0)

### 6.2.3 Wait Control Registers H and L (WCRH, WCRL)

WCRH and WCRL are 8-bit readable/writable registers that select the number of program wait states for each area.

In normal mode, only part of area is 0 is enabled, and bits W01 and W00 select the number of program wait states for the external space . The settings of bits W71, W70 to W11, and W10 have no effect on operation.

Program waits are not inserted in the case of on-chip memory or internal I/O registers.

WCRH and WCRL are initialized to H'FF by a power-on reset and in hardware standby mode. They are not initialized by a manual reset or in software standby mode.

**(1) WCRH**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Area 7 Wait Control 1 and 0 (W71, W70):** These bits select the number of program wait states when area 7 in external space is accessed while the AST7 bit in ASTCR is set to 1.

| Bit 7 | Bit 6 | |
|---|---|---|
| **W71** | **W70** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 7 is accessed |
| | 1 | 1 program wait state inserted when external space area 7 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 7 is accessed |
| | 1 | 3 program wait states inserted when external space area 7 is accessed (Initial value) |

**Bits 5 and 4—Area 6 Wait Control 1 and 0 (W61, W60):** These bits select the number of program wait states when area 6 in external space is accessed while the AST6 bit in ASTCR is set to 1.

| Bit 5 | Bit 4 | |
|---|---|---|
| **W61** | **W60** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 6 is accessed |
| | 1 | 1 program wait state inserted when external space area 6 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 6 is accessed |
| | 1 | 3 program wait states inserted when external space area 6 is accessed (Initial value) |

**Bits 3 and 2—Area 5 Wait Control 1 and 0 (W51, W50):** These bits select the number of program wait states when area 5 in external space is accessed while the AST5 bit in ASTCR is set to 1.

| Bit 3 | Bit 2 | |
|---|---|---|
| **W51** | **W50** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 5 is accessed |
| | 1 | 1 program wait state inserted when external space area 5 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 5 is accessed |
| | 1 | 3 program wait states inserted when external space area 5 is accessed (Initial value) |

**Bits 1 and 0—Area 4 Wait Control 1 and 0 (W41, W40):** These bits select the number of program wait states when area 4 in external space is accessed while the AST4 bit in ASTCR is set to 1.

| Bit 1 | Bit 0 | |
|---|---|---|
| **W41** | **W40** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 4 is accessed |
| | 1 | 1 program wait state inserted when external space area 4 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 4 is accessed |
| | 1 | 3 program wait states inserted when external space area 4 is accessed (Initial value) |

**(2) WCRL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Area 3 Wait Control 1 and 0 (W31, W30):** These bits select the number of program wait states when area 3 in external space is accessed while the AST3 bit in ASTCR is set to 1.

| Bit 7 | Bit 6 | |
|---|---|---|
| **W31** | **W30** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 3 is accessed |
| | 1 | 1 program wait state inserted when external space area 3 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 3 is accessed |
| | 1 | 3 program wait states inserted when external space area 3 is accessed (Initial value) |

**Bits 5 and 4—Area 2 Wait Control 1 and 0 (W21, W20):** These bits select the number of program wait states when area 2 in external space is accessed while the AST2 bit in ASTCR is set to 1.

| Bit 5 | Bit 4 | |
|---|---|---|
| **W21** | **W20** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 2 is accessed |
| | 1 | 1 program wait state inserted when external space area 2 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 2 is accessed |
| | 1 | 3 program wait states inserted when external space area 2 is accessed (Initial value) |

**Bits 3 and 2—Area 1 Wait Control 1 and 0 (W11, W10):** These bits select the number of program wait states when area 1 in external space is accessed while the AST1 bit in ASTCR is set to 1.

| Bit 3 | Bit 2 | |
|---|---|---|
| **W11** | **W10** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 1 is accessed |
| | 1 | 1 program wait state inserted when external space area 1 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 1 is accessed |
| | 1 | 3 program wait states inserted when external space area 1 is accessed (Initial value) |

**Bits 1 and 0—Area 0 Wait Control 1 and 0 (W01, W00):** These bits select the number of program wait states when area 0 in external space is accessed while the AST0 bit in ASTCR is set to 1.

| Bit 1 | Bit 0 | |
|---|---|---|
| **W01** | **W00** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 0 is accessed |
| | 1 | 1 program wait state inserted when external space area 0 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 0 is accessed |
| | 1 | 3 program wait states inserted when external space area 0 is accessed (Initial value) |

### 6.2.4 Bus Control Register H (BCRH)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | RMTS2 | RMTS1 | RMTS0 |
| Initial value | : | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BCRH is an 8-bit readable/writable register that selects enabling or disabling of idle cycle insertion, and the memory interface for areas 2 to 5 and area 0.

BCRH is initialized to H'D0 by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—Idle Cycle Insert 1 (ICIS1):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read cycles are performed in different areas.

**Bit 7**

| ICIS1 | Description |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read cycles in different areas |
| 1 | Idle cycle inserted in case of successive external read cycles in different areas (Initial value) |

**Bit 6—Idle Cycle Insert 0 (ICIS0):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read and external write cycles are performed .

**Bit 6**

| ICIS0 | Description |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read and external write cycles |
| 1 | Idle cycle inserted in case of successive external read and external write cycles (Initial value) |

**Bit 5—Burst ROM Enable (BRSTRM):** Selects whether area 0 is used as a burst ROM interface. In normal mode, the selection can be made from the entire external space .

Burst ROM interface and PSRAM burst operation cannot be set at the same time.

**Bit 5**

| BRSTRM | Description |
|---|---|
| 0 | Area 0 is basic bus interface (Initial value) |
| 1 | Area 0 is burst ROM interface |

**Bit 4—Burst Cycle Select 1 (BRSTS1):** Selects the number of burst cycles for the burst ROM interface.

| Bit 4 | |
|---|---|
| **BRSTS1** | **Description** |
| 0 | Burst cycle comprises 1 state |
| 1 | Burst cycle comprises 2 states (Initial value) |

**Bit 3—Burst Cycle Select 0 (BRSTS0):** Selects the number of words that can be accessed in a burst ROM interface burst access.

| Bit 3 | |
|---|---|
| **BRSTS0** | **Description** |
| 0 | Max. 4 words in burst access (Initial value) |
| 1 | Max. 8 words in burst access |

**Bits 2 to 0—RAM Type Select (RMTS2 to RMTS0):** These bits select the memory interface for areas 2 to 5 in advanced mode.

When DRAM space is selected, the relevant area is designated as DRAM interface, and when PSRAM space is selected, it is designated as PSRAM interface.

| Bit 2 | Bit 1 | Bit 0 | Description | | | |
|---|---|---|---|---|---|---|
| **RMTS2** | **RMTS1** | **RMTS0** | **Area 5** | **Area 4** | **Area 3** | **Area 2** |
| 0 | 0 | 0 | Normal space | | | |
| | | 1 | Normal space | | | DRAM space |
| | 1 | 0 | Normal space | | DRAM space | |
| | | 1 | DRAM space | | | |
| 1 | 0 | 0 | Normal space | | | |
| | | 1 | Normal space | | | PSRAM space |
| | 1 | 0 | Normal space | | PSRAM space | |
| | | 1 | PSRAM space | | | |

### 6.2.5 Bus Control Register L (BCRL)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | BRLE | BREQOE | EAE | LCASS | DDS | ASS | WDBE | WAITE |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BCRL is an 8-bit readable/writable register that performs selection of the external bus-released state protocol, the area partition unit, the LCAS signal, DMAC single address transfer, enabling or disabling of the write data buffer function, and enabling or disabling of $\overline{\text{WAIT}}$ pin input.

BCRL is initialized to H'3C by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—Bus Release Enable (BRLE):** Enables or disables external bus release.

**Bit 7**

| BRLE | Description |
|---|---|
| 0 | External bus release is disabled. $\overline{\text{BREQ}}$, $\overline{\text{BACK}}$, and $\overline{\text{BREQO}}$ can be used as I/O ports. (Initial value) |
| 1 | External bus release is enabled. |

**Bit 6—BREQO Pin Enable (BREQOE):** Outputs a signal that requests the external bus master to drop the bus request signal ($\overline{\text{BREQ}}$) in the external bus release state, when an internal bus master performs an external space access, or when a refresh request is generated.

**Bit 6**

| BREQOE | Description |
|---|---|
| 0 | $\overline{\text{BREQO}}$ output disabled. $\overline{\text{BREQO}}$ can be used as I/O port. (Initial value) |
| 1 | $\overline{\text{BREQO}}$ output enabled. |

**Bit 5—External Address Enable (EAE):** Selects whether addresses H'010000 to H'01FFFF are to be internal addresses or external addresses.

This setting is invalid in normal mode.

**Bit 5**

| EAE | Description |
|---|---|
| 0 | Addresses H'010000 to H'01FFFF are in on-chip ROM (in the H8S/2655) or a reserved area* (in the H8S/2653) |
| 1 | Addresses H'010000 to H'01FFFF are external addresses (external expansion mode) or a reserved area* (single-chip mode) (Initial value) |

Note: * Reserved areas should not be accessed.

**Bit 4—LCAS Select (LCASS):** Selects use of the $\overline{\text{LWR}}$ pin or the $\overline{\text{LCAS}}$ pin for the 2-CAS type DRAM interface $\overline{\text{LCAS}}$ signal.

**Bit 4**

| LCASS | Description |
|---|---|
| 0 | $\overline{\text{LCAS}}$ pin used for 2-CAS type DRAM interface $\overline{\text{LCAS}}$ signal ($\overline{\text{BREQO}}$ output and $\overline{\text{WAIT}}$ input cannot be used when $\overline{\text{LCAS}}$ signal is used) |
| 1 | $\overline{\text{LWR}}$ pin used for 2-CAS type DRAM interface $\overline{\text{LCAS}}$ signal (RAS down mode cannot be used) (Initial value) |

**Bit 3—DACK Timing Select (DDS):** Selects the DMAC single address transfer bus timing for the DRAM interface or PSRAM interface.

**Bit 3**

| DDS | Description |
|---|---|
| 0 | When DMAC single address transfer is performed in DRAM/PSRAM space, full access is always executed<br>$\overline{\text{DACK}}$ signal goes low from $T_r$ or $T_1$ cycle |
| 1 | Burst access is possible when DMAC single address transfer is performed in DRAM/PSRAM space<br>$\overline{\text{DACK}}$ signal goes low from $T_{c1}$ or $T_2$ cycle (Initial value) |

**Bit 2—Area Partition Unit Select (ASS):** Selects the area partition unit.

**Bit 2**

| ASS | Description | |
|---|---|---|
| 0 | Area partition unit is 128 kbytes (1 Mbit) | |
| 1 | Area partition unit is 2 Mbytes (16 Mbits) | (Initial value) |

**Bit 1—Write Data Buffer Enable (WDBE):** Selects whether or not the write buffer function is used for an external write cycle or DMAC single address cycle.

**Bit 1**

| WDBE | Description | |
|---|---|---|
| 0 | Write data buffer function not used | (Initial value) |
| 1 | Write data buffer function used | |

**Bit 0—WAIT Pin Enable (WAITE):** Selects enabling or disabling of wait input by the $\overline{\text{WAIT}}$ pin.

**Bit 0**

| WAITE | Description | |
|---|---|---|
| 0 | Wait input by $\overline{\text{WAIT}}$ pin disabled. $\overline{\text{WAIT}}$ pin can be used as I/O port. | (Initial value) |
| 1 | Wait input by $\overline{\text{WAIT}}$ pin enabled | |

### 6.2.6 Memory Control Register (MCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TPC | BE | RCDM | CW2 | MXC1 | MXC0 | RLW1 | RLW0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MCR is an 8-bit readable/writable register that selects the DRAM strobe control method, number of precharge cycles, access mode, address multiplexing shift size, and the number of wait states inserted during refreshing, when areas 2 to 5 are designated as DRAM interface. When areas 2 to 5 are designated as PSRAM interface, MCR selects the number of precharge cycles and the access mode for PSRAM.

MCR is initialized to H'00 by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—TP Cycle Control (TPC):** Selects whether a 1-state or 2-state precharge cycle ($T_P$) is to be used when areas 2 to 5 designated as DRAM space or PSRAM space are accessed.

**Bit 7**

| TPC | Description | |
|---|---|---|
| 0 | 1-state precharge cycle is inserted | (Initial value) |
| 1 | 2-state precharge cycle is inserted | |

**Bit 6—Burst Access Enable (BE):** Selects enabling or disabling of burst access to areas 2 to 5 designated as DRAM space or PSRAM space. DRAM space burst access is performed in fast page mode, and PSRAM access in static column mode.

PSRAM burst operation and burst ROM interface cannot be set at the same time.

**Bit 6**

| BE | Description | |
|---|---|---|
| 0 | Burst disabled (always full access) | (Initial value) |
| 1 | • For DRAM space access<br>Access in fast page mode<br>• For PSRAM space access<br>Access in static column mode | |

**Bit 5—RAS Down Mode (RCDM):** When areas 2 to 5 are designated as DRAM space and access to DRAM is interrupted, RCDM selects whether the next DRAM access is waited for with the $\overline{RAS}$ signal held low (RAS down mode), or the $\overline{RAS}$ signal is driven high again (RAS up mode). RAS down mode cannot be used with the 2-CAS method (LCASS=1). When selecting RAS down mode, set the BE bit to 1.

When areas 2 to 5 are designated as PSRAM space, this bit is invalid.

**Bit 5**

| RCDM | Description | |
|---|---|---|
| 0 | DRAM interface: RAS up mode selected | (Initial value) |
| 1 | DRAM interface: RAS down mode selected | |

**Bit 4—2-CAS Method/2-WE Method Select (CW2):** Selects whether the 2-CAS method or 2-WE method is used for byte access when areas 2 to 5 are designated as 16-bit DRAM space.

**Bit 4**

| CW2 | Description | |
|---|---|---|
| 0 | 2-CAS method selected: $\overline{CASH}$, $\overline{CASL}$, $\overline{WE}$ signals enabled | (Initial value) |
| 1 | 2-WE method selected: $\overline{CAS}$, $\overline{UWE}$, $\overline{LWE}$ signals enabled | |

**Bits 3 and 2—Multiplex Shift Count 1 and 0 (MXC1, MXC0):** These bits select the size of the shift to the lower half of the row address in row address/column address multiplexing for the DRAM interface. In burst operation on the DRAM/PSRAM interface, these bits also select the row address to be used for comparison.

| Bit 3 | Bit 2 | |
|---|---|---|
| **MXC1** | **MXC0** | **Description** |
| 0 | 0 | 8-bit shift (Initial value)<br>• When 8-bit access space is designated: Row address $A_{23}$ to $A_8$ used for comparison<br>• When 16-bit access space is designated: Row address $A_{23}$ to $A_9$ used for comparison |
| | 1 | 9-bit shift<br>• When 8-bit access space is designated: Row address $A_{23}$ to $A_9$ used for comparison<br>• When 16-bit access space is designated: Row address $A_{23}$ to $A_{10}$ used for comparison |
| 1 | 0 | 10-bit shift<br>• When 8-bit access space is designated: Row address $A_{23}$ to $A_{10}$ used for comparison<br>• When 16-bit access space is designated: Row address $A_{23}$ to $A_{11}$ used for comparison |
| | 1 | — |

**Bits 1 and 0—Refresh Cycle Wait Control 1 and 0 (RLW1, RLW0):** These bits select the number of wait states to be inserted in a DRAM interface CAS-before-RAS refresh cycle. This setting is used for all areas designated as DRAM space. Wait input by the $\overline{\text{WAIT}}$ pin is disabled.

| Bit 1 | Bit 0 | |
|---|---|---|
| **RLW1** | **RLW0** | **Description** |
| 0 | 0 | No wait state inserted (Initial value) |
| | 1 | 1 wait state inserted |
| 1 | 0 | 2 wait states inserted |
| | 1 | 3 wait states inserted |

### 6.2.7 DRAM Control Register (DRAMCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | RFSHE | RCW | RMODE | CMF | CMIE | CKS2 | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DRAMCR is an 8-bit readable/writable register that selects the DRAM refresh mode and refresh counter clock, and controls the refresh timer.

DRAMCR is initialized to H'00 by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—Refresh Control (RFSHE):** Selects whether or not refresh control is performed. When refresh control is not performed, the refresh timer can be used as an interval timer. Refresh control is not performed in normal mode.

**Bit 7**

| RFSHE | Description | |
|---|---|---|
| 0 | Refresh control is not performed | (Initial value) |
| 1 | Refresh control is performed | |

**Bit 6—RAS-CAS Wait (RCW):** Controls wait state insertion in DRAM interface CAS-before-RAS refreshing. In case of the PSRAM interface, the value of this bit should be kept at 0.

**Bit 6**

| RCW | Description | |
|---|---|---|
| 0 | Wait state insertion in CAS-before-RAS refreshing disabled<br>$\overline{RAS}$ falls in $T_{Rr}$ cycle | (Initial value) |
| 1 | One wait state inserted in CAS-before-RAS refreshing<br>$\overline{RAS}$ falls in $T_{Rc1}$ cycle | |

**Bit 5—Refresh Mode (RMODE):** When refresh control is performed (RFSHE = 1), this bit selects whether normal refreshing (CAS-before-RAS refreshing for the DRAM interface, auto-refreshing for the PSRAM interface) or self-refreshing is performed.

**Bit 5**

| RMODE | Description |
|---|---|
| 0 | • DRAM interface<br>CAS-before-RAS refreshing used<br>• PSRAM interface<br>Auto-refreshing used (Initial value) |
| 1 | Self-refreshing used |

**Bit 4—Compare Match Flag (CMF):** Status flag that indicates a match between the values of RTCNT and RTCOR.

When refresh control is performed (RFSHE = 1), 1 should be written to the CMF bit when writing to DRAMCR.

**Bit 4**

| CMF | Description |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading the CMF flag when CMF = 1, then writing 0 to the CMF flag (Initial value) |
| 1 | [Setting condition]<br>Set when RTCNT = RTCOR |

**Bit 3—Compare Match Interrupt Enable (CMIE):** Enables or disables interrupt requests (CMI) by the CMF flag when the CMF flag in DRAMCR is set to 1.

When refresh control is performed (RFSHE = 1), the CMIE bit is always cleared to 0.

**Bit 3**

| CMIE | Description |
|---|---|
| 0 | Interrupt request (CMI) by CMF flag disabled (Initial value) |
| 1 | Interrupt request (CMI) by CMF flag enabled |

**Bits 2 to 0—Refresh Counter Clock Select (CKS2 to CKS0):** These bits select the clock to be input to RTCNT from among 7 internal clocks obtained by dividing the system clock (ø). When the input clock is selected with bits CKS2 to CKS0, RTCNT begins counting up.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| **CKS2** | **CKS1** | **CKS0** | **Description** | |
| 0 | 0 | 0 | Count operation disabled | (Initial value) |
| | | 1 | Count uses ø/2 | |
| | 1 | 0 | Count uses ø/8 | |
| | | 1 | Count uses ø/32 | |
| 1 | 0 | 0 | Count uses ø/128 | |
| | | 1 | Count uses ø/512 | |
| | 1 | 0 | Count uses ø/2048 | |
| | | 1 | Count uses ø/4096 | |

### 6.2.8 Refresh Timer/Counter (RTCNT)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

RTCNT is an 8-bit readable/writable up-counter.

RTCNT counts up using the internal clock selected by bits CKS2 to CKS0 in DRAMCR.

When RTCNT matches RTCOR (compare match), the CMF flag in DRAMCR is set to 1 and RTCNT is cleared to H'00. If the RFSHE bit in DRAMCR is set to 1 at this time, a refresh cycle is started. Also, if the CMIE bit in DRAMCR is set to 1, a compare match interrupt (CMI) is generated.

RTCNT is initialized to H'00 by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

### 6.2.9 Refresh Time Constant Register (RTCOR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

RTCOR is an 8-bit readable/writable register that sets the period for compare match operations with RTCNT.

The values of RTCOR and RTCNT are constantly compared, and if they match, the CMF flag in DRAMCR is set to 1 and RTCNT is cleared to H'00.

RTCOR is initialized to H'FF by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

## 6.3 Overview of Bus Control

### 6.3.1 Area Partitioning

In advanced mode, the bus controller partitions the 16 Mbytes address space into eight areas, 0 to 7, in 128-kbyte or 2-Mbyte units, and performs bus control for external space in area units. In normal mode, it controls a 64-kbyte address space comprising part of area 0. Figure 6-2 shows an outline of the memory map.

Chip select signals ($\overline{CS}_0$ to $\overline{CS}_7$) can be output for each area.

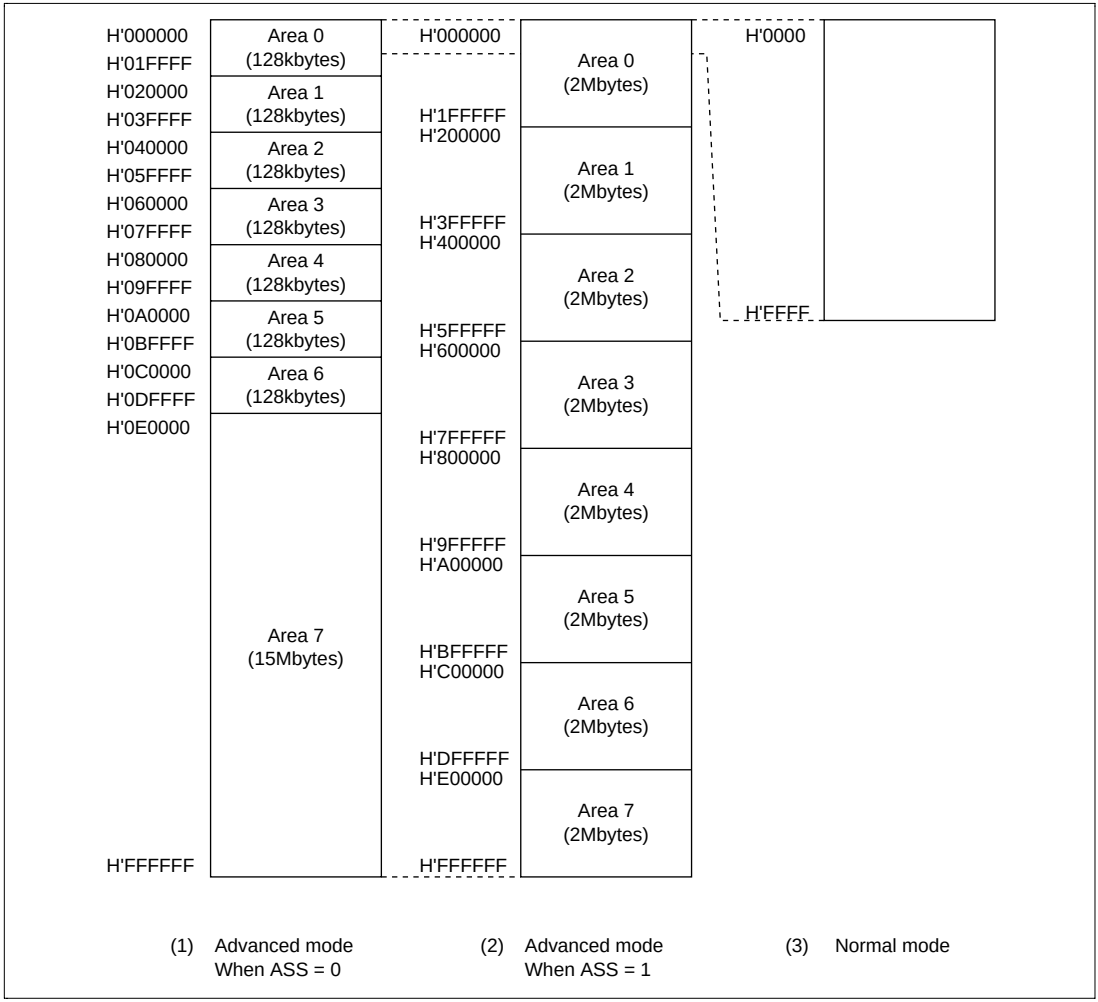



**Figure 6-2 Overview of Area Partitioning**

### 6.3.2 Bus Specifications

The external space bus specifications consist of three elements: bus width, number of access states, and number of program wait states.

The bus width and number of access states for on-chip memory and internal I/O registers are fixed, and are not affected by the bus controller.

**(1) Bus Width:** A bus width of 8 or 16 bits can be selected with ADWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space, and an area for which a 16-bit bus is selected functions as a16-bit access space.

If all areas are designated for 8-bit access, 8-bit bus mode is set; if any area is designated for 16-bit access, 16-bit bus mode is set. When the burst ROM interface is designated, 16-bit bus mode is always set.

**(2) Number of Access States:** Two or three access states can be selected with ASTCR. An area for which 2-state access is selected functions as a 2-state access space, and an area for which 3-state access is selected functions as a 3-state access space.

With the DRAM/PSRAM interface and burst ROM interface, the number of access states may be determined without regard to ASTCR.

When 2-state access space is designated, wait insertion is disabled.

**(3) Number of Program Wait States:** When 3-state access space is designated by ASTCR, the number of program wait states to be inserted automatically is selected with WCRH and WCRL. From 0 to 3 program wait states can be selected.

Table 6-3 shows the bus specifications for each basic bus interface area.

**Table 6-3 Bus Specifications for Each Area (Basic Bus Interface)**

| ABWCR | ASTCR | WCRH, WCRL | | Bus Specifications (Basic Bus Interface) | | |
|---|---|---|---|---|---|---|
| ABWn | ASTn | Wn1 | Wn0 | Bus Width | Access States | Program Wait States |
| 0 | 0 | — | — | 16 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |
| 1 | 0 | — | — | 8 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |

### 6.3.3 Memory Interfaces

The H8S/2655 Series memory interfaces comprise a basic bus interface that allows direct connection of ROM, SRAM, and so on; a DRAM interface that allows direct connection of DRAM; a PSRAM interface that allows direct connection of PSRAM; and a burst ROM interface that allows direct connection of burst ROM. The interface can be selected independently for each area.

An area for which the basic bus interface is designated functions as normal space, an area for which the DRAM interface is designated functions as DRAM space, an area for which the PSRAM interface is designated functions as PSRAM space, and an area for which the burst ROM interface is designated functions as burst ROM space.

### 6.3.4 Advanced Mode

The initial state of each area is basic bus interface, 3-state access space. The initial bus width is selected according to the operating mode. The bus specifications described here cover basic items only, and the sections on each memory interface (6.5, 6.6, and 6.7) should be referred to for further details.

**Area 0:** Area 0 includes on-chip ROM, and in ROM-disabled expansion mode, all of area 0 is external space. In ROM-enabled expansion mode, the space excluding on-chip ROM is external space.

When area 0 external space is accessed, the $\overline{CS}_0$ signal can be output.

Either basic bus interface or burst ROM interface can be selected for area 0.

The size of area 0 is switched between 128 kbytes and 2 Mbytes according to the state of the ASS bit.

**Areas 1 and 6:** In external expansion mode, all of areas 1 and 6 is external space.

When area 1 and 6 external space is accessed, the $\overline{CS}_1$ and $\overline{CS}_6$ pin signals respectively can be output.

Only the basic bus interface can be used for areas 1 and 6.

The size of areas 1 and 6 is switched between 128 kbytes and 2 Mbytes according to the state of the ASS bit.

**Areas 2 to 5:** In external expansion mode, all of areas 2 to 5 is external space.

When area 2 to 5 external space is accessed, signals $\overline{CS}_2$ to $\overline{CS}_5$ can be output.

Basic bus interface, DRAM interface, or PSRAM interface can be selected for areas 2 to 5. With the DRAM interface, signals $\overline{CS}_2$ to $\overline{CS}_5$ are used as $\overline{RAS}$ signals.

The size of areas 2 to 5 is switched between 128 kbytes and 2 Mbytes according to the state of the ASS bit.

**Area 7:** Area 7 includes the on-chip RAM and internal I/O registers. In external expansion mode, the space excluding the on-chip RAM and internal I/O registers is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space .

When area 7 external space is accessed, the $\overline{CS}_7$ signal can be output.

Only the basic bus interface can be used for the area 7 memory interface.

The size of area 7 is switched between 15 Mbytes and 2 Mbytes according to the state of the ASS bit.

### 6.3.5 Areas in Normal Mode

In normal mode, a 64-kbyte address space comprising part of area 0 is controlled. Area partitioning is not performed in normal mode. In ROM-disabled expansion mode, the space excluding the on-chip RAM and internal I/O registers is external space. In ROM-enabled expansion mode the space excluding the on-chip ROM, on-chip RAM, and internal I/O registers is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space .

When external space is accessed, the $\overline{CS}_0$ signal can be output.

The basic bus interface or burst ROM interface can be selected.

### 6.3.6 Chip Select Signals

The H8S/2655 Series can output chip select signals ($\overline{CS}_0$ to $\overline{CS}_7$) to areas 0 to 7, the signal being driven low when the corresponding external space area is accessed. In normal mode, only the $\overline{CS}_0$ signal can be output.

Figure 6-3 shows an example of $\overline{CS}_n$ (n = 0 to 7) output timing.

Enabling or disabling of the $\overline{CS}_n$ signal is performed by setting the data direction register (DDR) for the port corresponding to the particular $\overline{CS}_n$ pin.

In ROM-disabled expansion mode, the $\overline{CS}_0$ pin is placed in the output state after a power-on reset. Pins $\overline{CS}_1$ to $\overline{CS}_7$ are placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS}_1$ to $\overline{CS}_7$.

In ROM-enabled expansion mode, pins $\overline{CS}_0$ to $\overline{CS}_7$ are all placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS}_0$ to $\overline{CS}_7$.

For details, see section 9, I/O Ports.

When areas 2 to 5 are designated as DRAM space, outputs $\overline{CS}_2$ to $\overline{CS}_5$ are used as $\overline{RAS}$ signals.

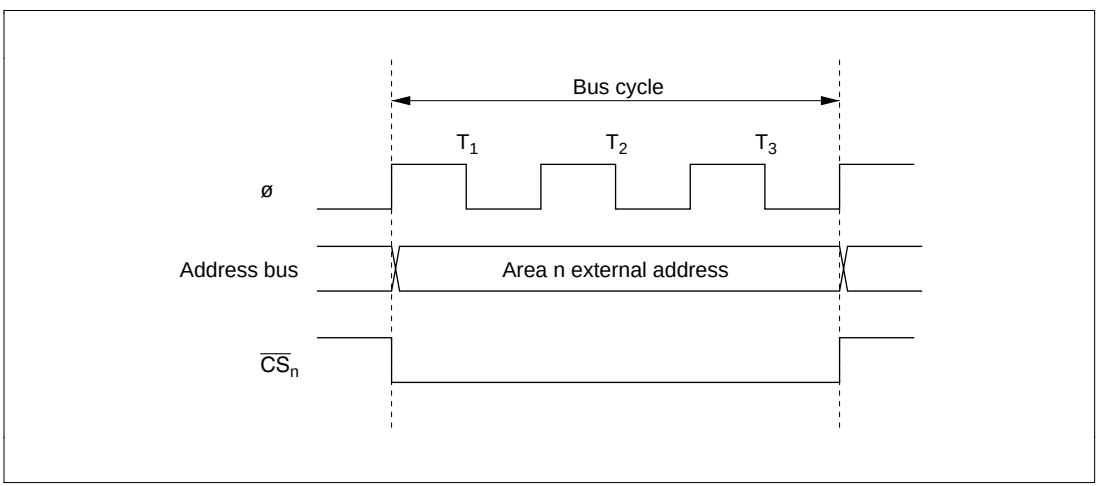



**Figure 6-3 $\overline{CS}_n$ Signal Output Timing (n = 0 to 7)**

## 6.4 Basic Bus Interface

### 6.4.1 Overview

The basic bus interface enables direct connection of ROM, SRAM, and so on.

The bus specifications can be selected with ABWCR, ASTCR, WCRH, and WCRL (see table 6-3).

### 6.4.2 Data Size and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and when accessing external space, controls whether the upper data bus ($D_{15}$ to $D_8$) or lower data bus ($D_7$ to $D_0$) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space) and the data size.

**8-Bit Access Space:** Figure 6-4 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus ($D_{15}$ to $D_8$) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word transfer instruction is performed as two byte accesses, and a longword transfer instruction, as four byte accesses.

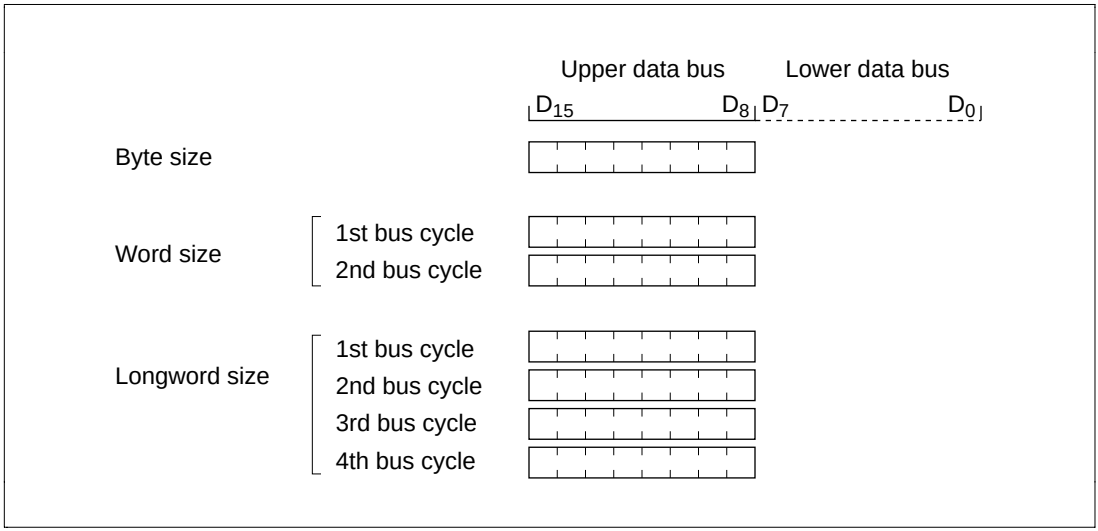



**Figure 6-4 Access Sizes and Data Alignment Control (8-Bit Access Space)**

**16-Bit Access Space:** Figure 6-5 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus ($D_{15}$ to $D_8$) and lower data bus ($D_7$ to $D_0$) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword transfer instruction is executed as two word transfer instructions.

In byte access, whether the upper or lower data bus is used is determined by whether the address is even or odd. The upper data bus is used for an even address, and the lower data bus for an odd address.

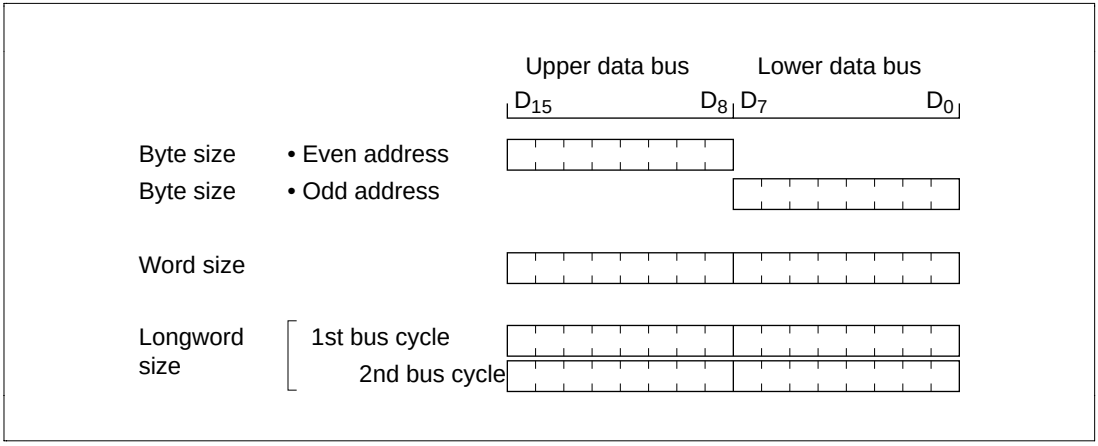



**Figure 6-5 Access Sizes and Data Alignment Control (16-Bit Access Space)**

### 6.4.3 Valid Strobes

Table 6-4 shows the data buses used and valid strobes for the access spaces.

In a read, the $\overline{RD}$ signal is valid without discrimination between the upper and lower halves of the data bus.

In a write, the $\overline{HWR}$ signal is valid for the upper half of the data bus, and the $\overline{LWR}$ signal for the lower half.

**Table 6-4 Data Buses Used and Valid Strobes**

| Area | Access Size | Read/ Write | Address | Valid Strobe | Upper Data Bus ($D_{15}$ to $D_8$) | Lower data bus ($D_7$ to $D_0$) |
|---|---|---|---|---|---|---|
| 8-bit access space | Byte | Read | — | $\overline{RD}$ | Valid | Invalid |
| | | Write | — | $\overline{HWR}$ | | Undefined |
| 16-bit access space | Byte | Read | Even | $\overline{RD}$ | Valid | Invalid |
| | | | Odd | | Invalid | Valid |
| | | Write | Even | $\overline{HWR}$ | Valid | Undefined |
| | | | Odd | $\overline{LWR}$ | Undefined | Valid |
| | Word | Read | — | $\overline{RD}$ | Valid | Valid |
| | | Write | — | $\overline{HWR}$, $\overline{LWR}$ | Valid | Valid |

Note: Undefined: Undefined data is output.
Invalid: Input state; input value is ignored.

### 6.4.4 Basic Timing

**8-Bit 2-State Access Space:** Figure 6-6 shows the bus timing for an 8-bit 2-state access space. When an 8-bit access space is accessed , the upper half ($D_{15}$ to $D_8$) of the data bus is used.

The $\overline{LWR}$ pin is fixed high. Wait states cannot be inserted.

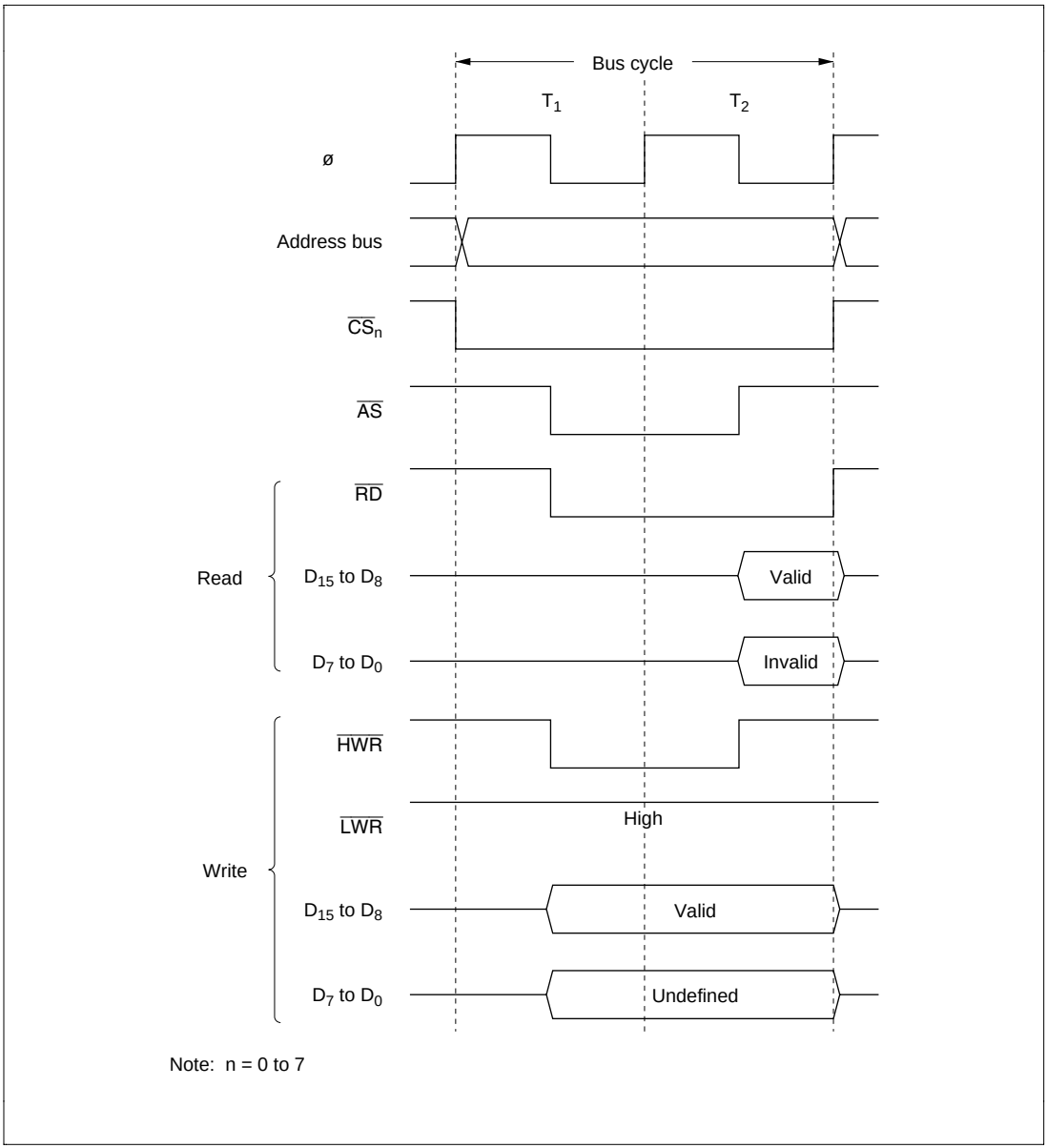



**Figure 6-6 Bus Timing for 8-Bit 2-State Access Space**

**8-Bit 3-State Access Space:** Figure 6-7 shows the bus timing for an 8-bit 3-state access space. When an 8-bit access space is accessed, the upper half ($D_{15}$ to $D_8$) of the data bus is used.

The $\overline{LWR}$ pin is fixed high. Wait states can be inserted.

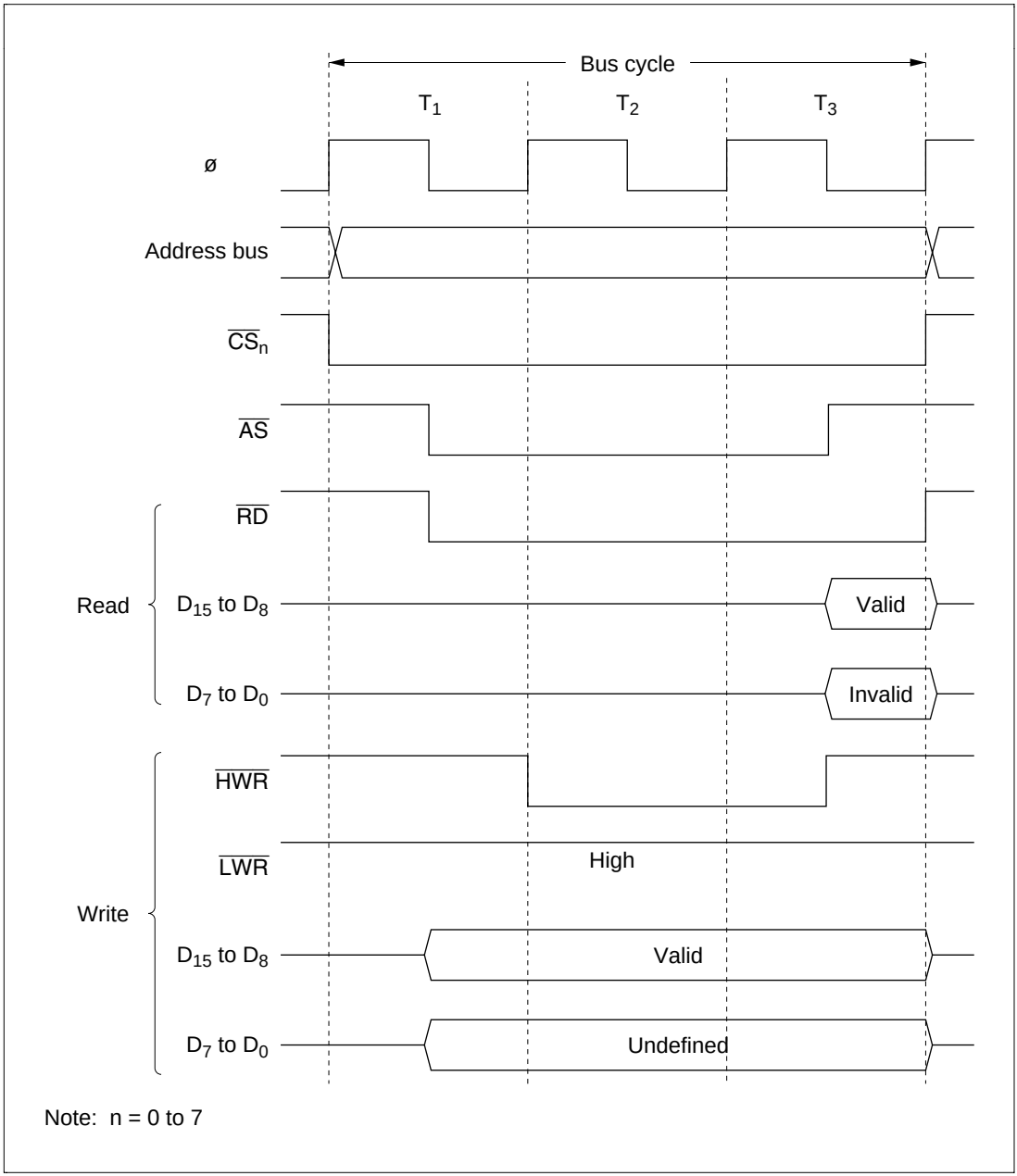



**Figure 6-7 Bus Timing for 8-Bit 3-State Access Space**

**16-Bit 2-State Access Space:** Figures 6-8 to 6-10 show bus timings for a 16-bit 2-state access space. When a 16-bit access space is accessed, the upper half ($D_{15}$ to $D_8$) of the data bus is used for the even address, and the lower half ($D_7$ to $D_0$) for the odd address.

Wait states cannot be inserted.

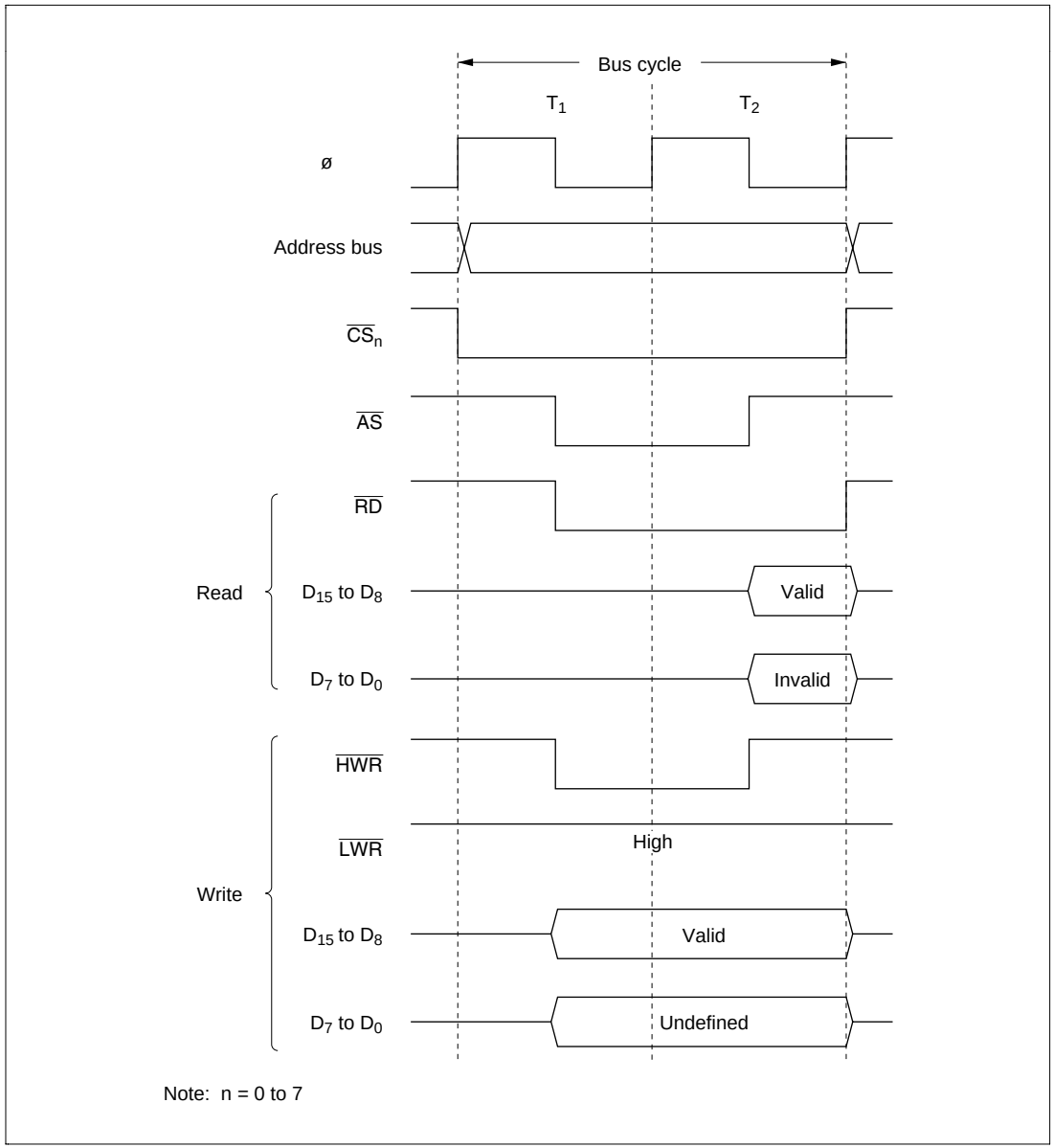



**Figure 6-8 Bus Timing for 16-Bit 2-State Access Space (1) (Even Address Byte Access)**

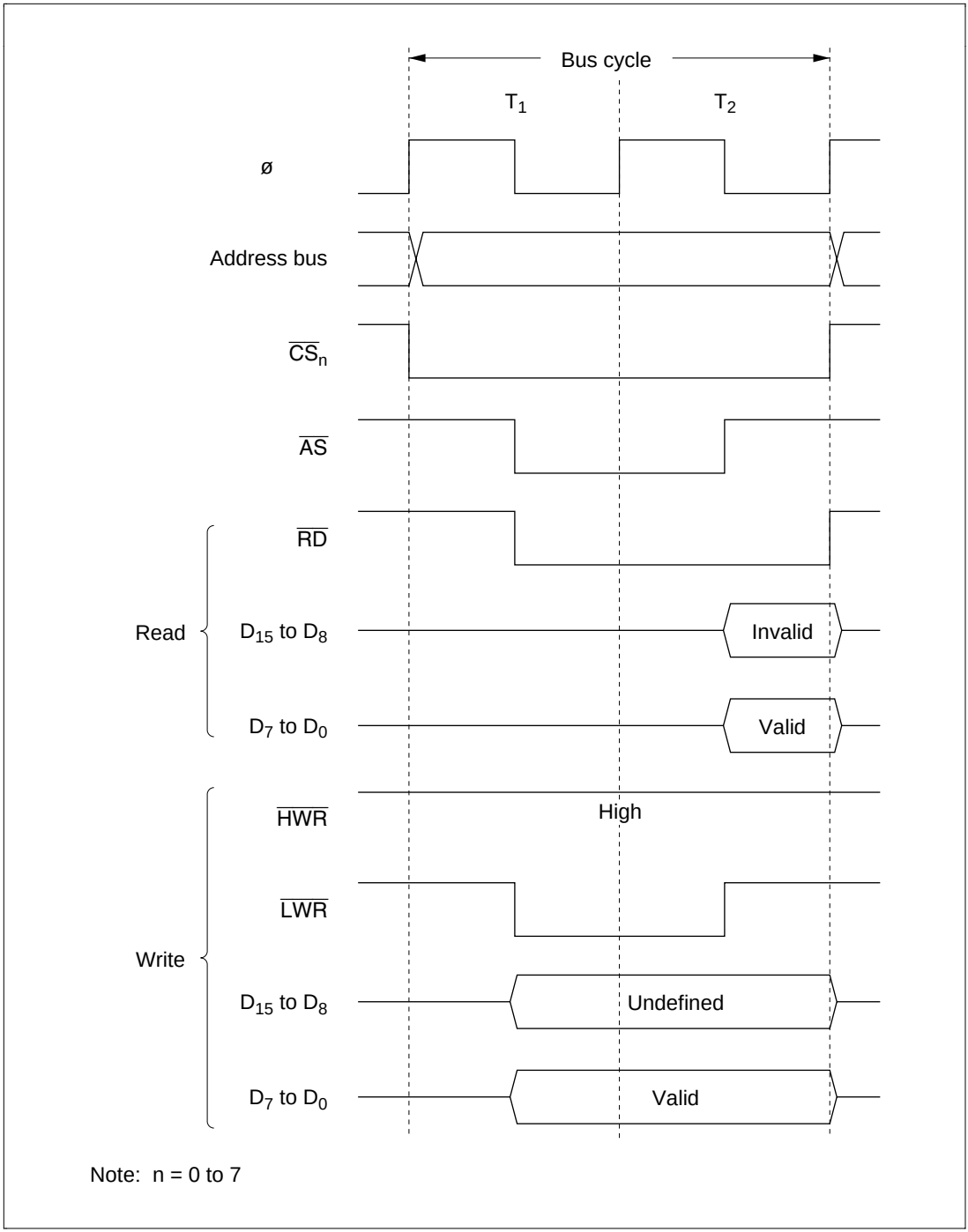
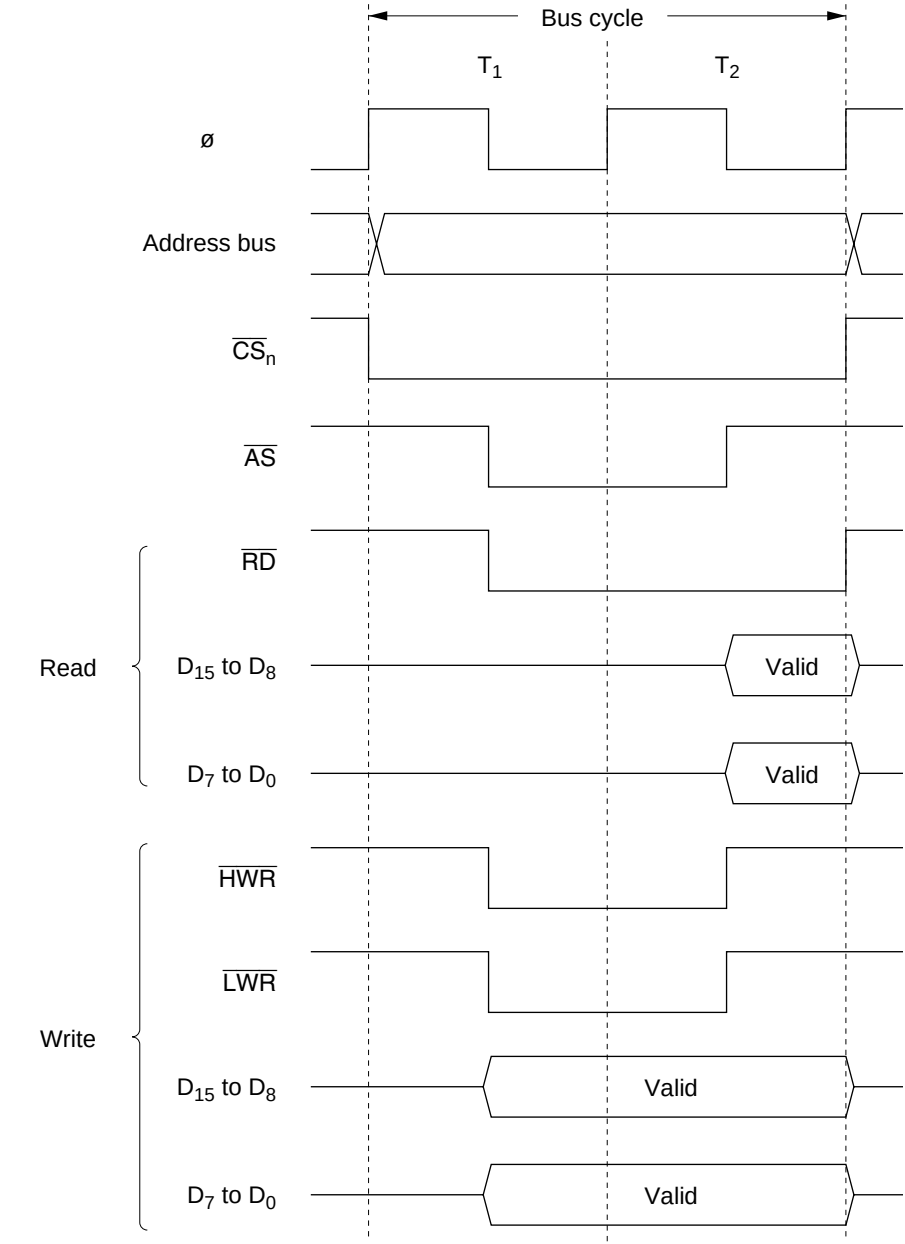



**Figure 6-9 Bus Timing for 16-Bit 2-State Access Space (2) (Odd Address Byte Access)**




Note: n = 0 to 7

**Figure 6-10 Bus Timing for 16-Bit 2-State Access Space (3) (Word Access)**

**16-Bit 3-State Access Space:** Figures 6-11 to 6-13 show bus timings for a 16-bit 3-state access space. When a 16-bit access space is accessed , the upper half ($D_{15}$ to $D_8$) of the data bus is used for the even address, and the lower half ($D_7$ to $D_0$) for the odd address.

Wait states can be inserted.

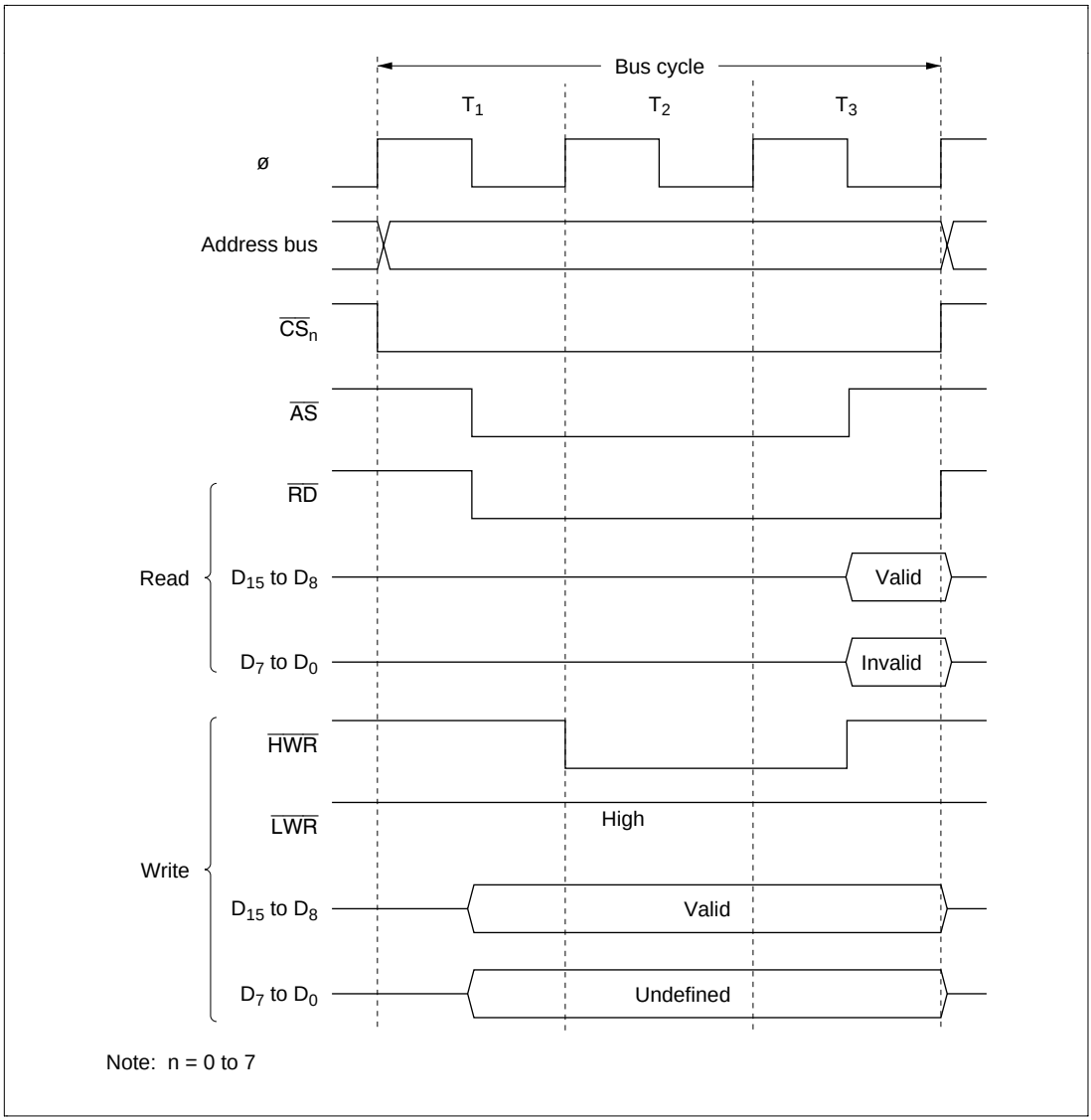



**Figure 6-11 Bus Timing for 16-Bit 3-State Access Space (1) (Even Address Byte Access)**

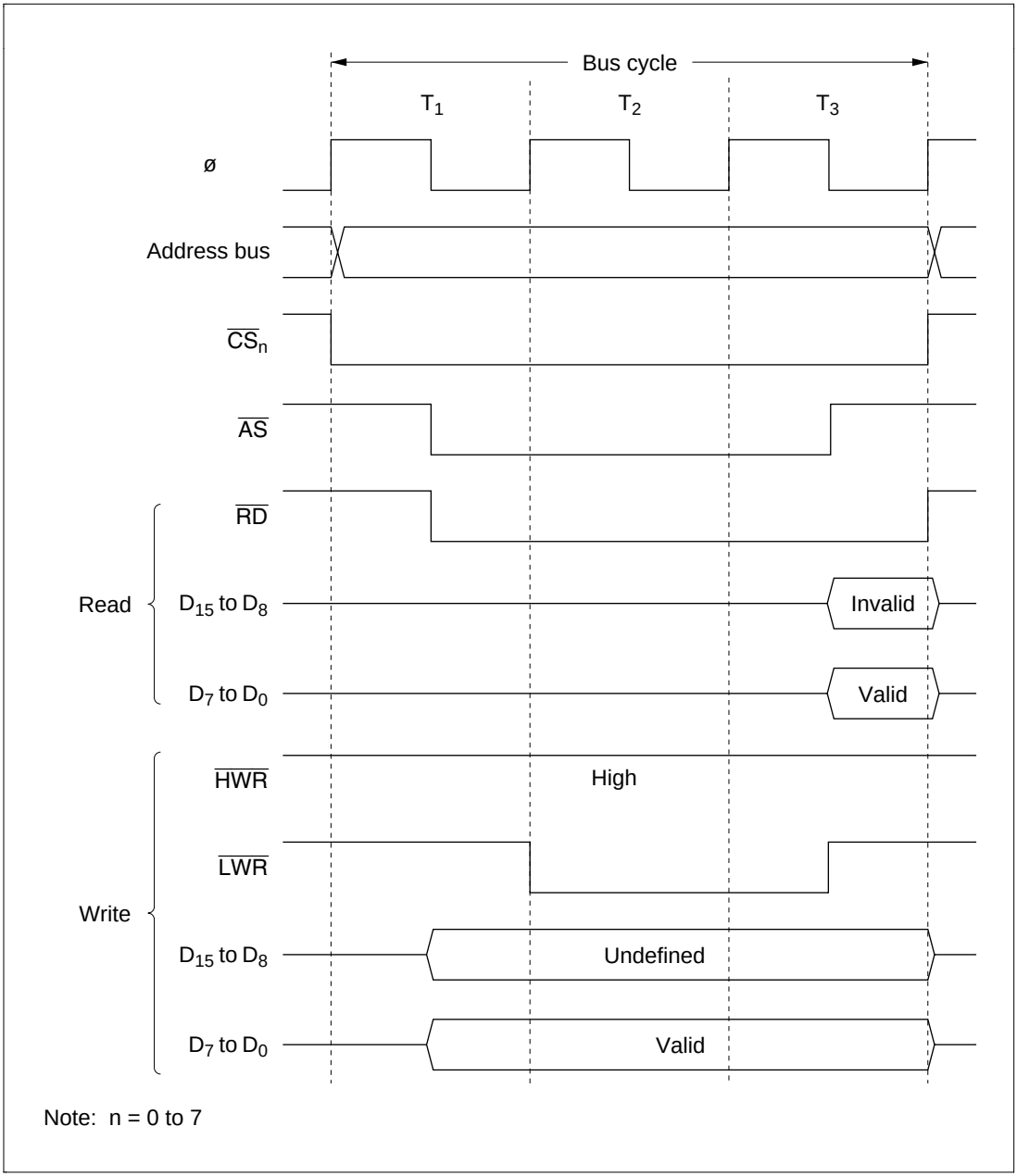



**Figure 6-12 Bus Timing for 16-Bit 3-State Access Space (2) (Odd Address Byte Access)**

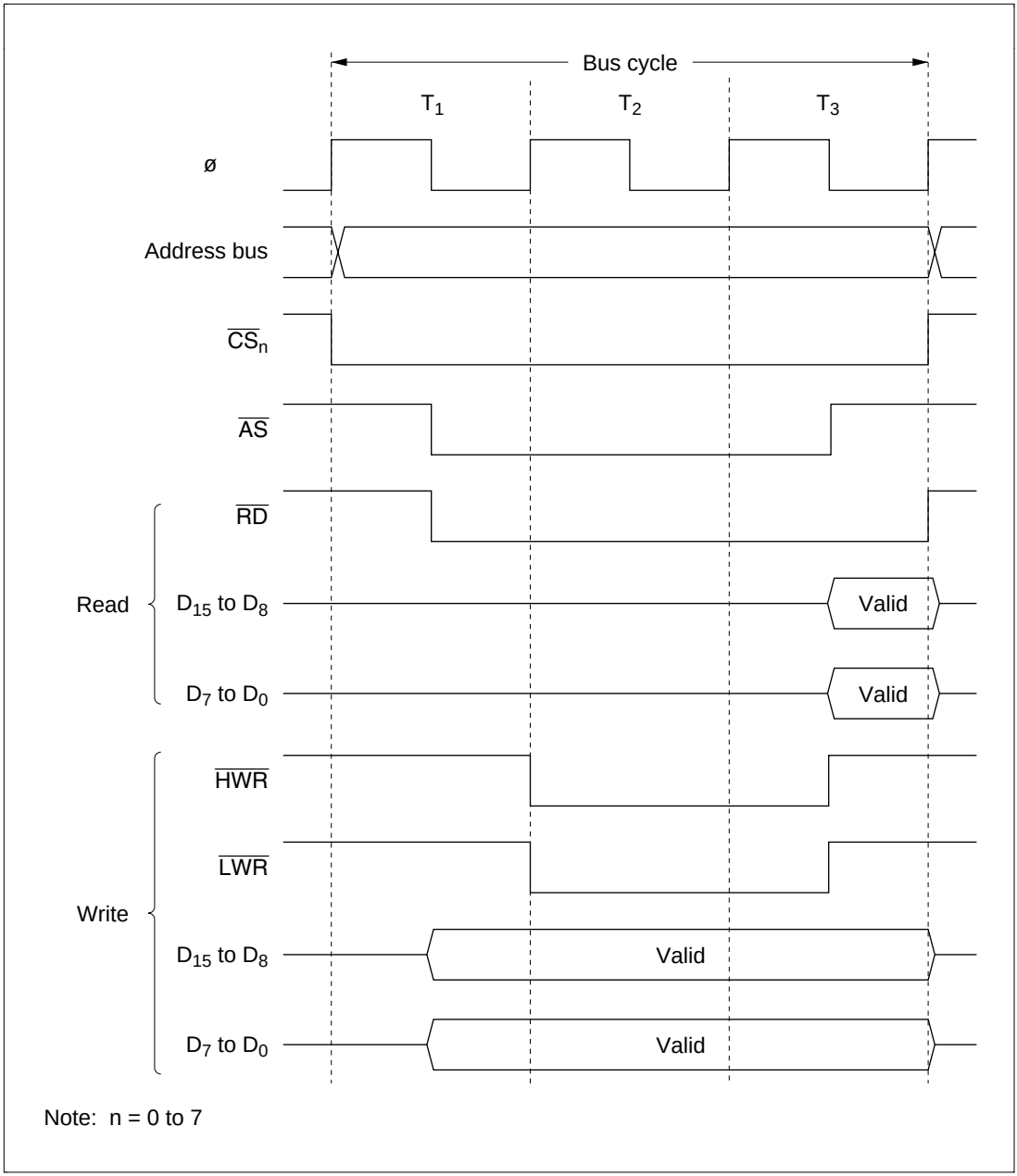



**Figure 6-13 Bus Timing for 16-Bit 3-State Access Space (3) (Word Access)**

### 6.4.5 Wait Control

When accessing external space, the H8S/2655 Series can extend the bus cycle by inserting one or more wait states ($T_w$). There are two ways of inserting wait states: program wait insertion and pin wait insertion using the $\overline{\text{WAIT}}$ pin.

**Program Wait Insertion**

From 0 to 3 wait states can be inserted automatically between the $T_2$ state and $T_3$ state on an individual area basis in 3-state access space, according to the settings of BWCRH and BWCRL.

**Pin Wait Insertion**

Setting the WAITE bit in BCRH to 1 enables wait insertion by means of the $\overline{\text{WAIT}}$ pin. When external space is accessed in this state, a program wait is first inserted. If the $\overline{\text{WAIT}}$ pin is low at the falling edge of ø in the last $T_2$ or $T_w$ state, another $T_w$ state is inserted. If the $\overline{\text{WAIT}}$ pin is held low, $T_w$ states are inserted until it goes high.

This is useful when inserting four or more $T_w$ states, or when changing the number of $T_w$ states for different external devices.

The WAITE bit setting applies to all areas.

Figure 6-14 shows an example of wait state insertion timing.

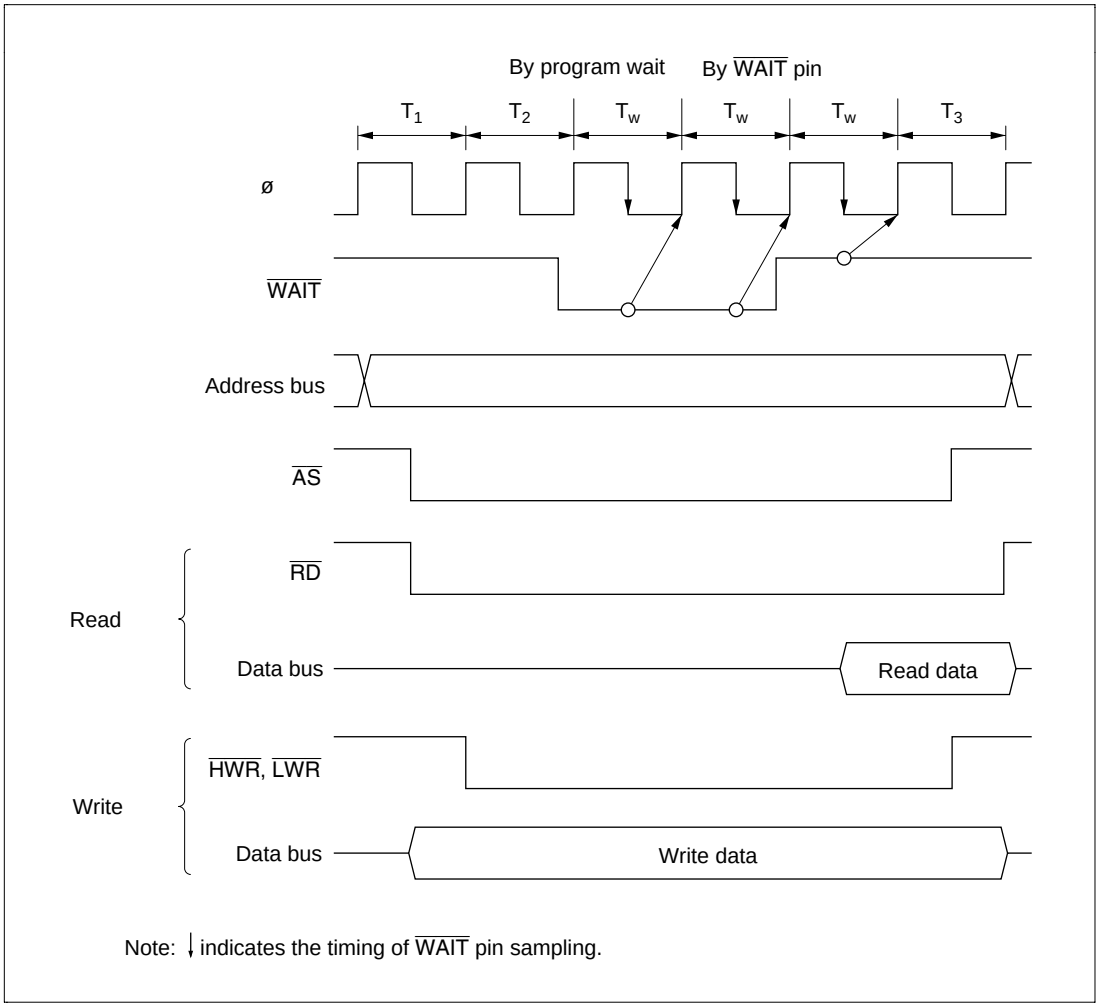



**Figure 6-14 Example of Wait State Insertion Timing**

The settings after a power-on reset are: 3-state access, 3 program wait state insertion, and WAIT input disabled. When a manual reset is performed, the contents of bus controller registers are retained, and the wait control settings remain the same as before the reset.

## 6.5 DRAM Interface

### 6.5.1 Overview

When the H8S/2655 Series is in advanced mode, external space areas 2 to 5 can be designated as DRAM space, and DRAM interfacing performed. With the DRAM interface, DRAM can be directly connected to the H8S/2655 Series. A DRAM space of 2, 4, or 8 Mbytes, or 128, 256, or 512 kbytes can be set by means of bits RMTS2 to RMTS0 in BCRH. Burst operation is also possible, using fast page mode.

### 6.5.2 Setting DRAM Space

Areas 2 to 5 are designated as DRAM space by setting bits RMTS2 to RMTS0 in BCRH. The relation between the settings of bits RMTS2 to RMTS0 and DRAM space is shown in table 6-5. Possible DRAM space settings are: one area (area 2), two areas (areas 2 and 3), and four areas (areas 2 to 5).

**Table 6-5 Settings of Bits RMTS2 to RMTS0 and Corresponding DRAM Spaces**

| RMTS2 | RMTS1 | RMTS0 | Area 5 | Area 4 | Area 3 | Area 2 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | Normal space | | | DRAM space |
| | 1 | 0 | Normal space | | DRAM space | |
| | | 1 | DRAM space | | | |

### 6.5.3 Address Multiplexing

With DRAM space, the row address and column address are multiplexed. In address multiplexing, the size of the shift of the row address is selected with bits MXC1 and MXC0 in MCR. Table 6-6 shows the relation between the settings of MXC1 and MXC0 and the shift size.

**Table 6-6 Address Multiplexing Settings by Bits MXC1 and MXC0**

| | MCR | | Shift | Address Pins | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MXC1 | MXC0 | Size | $A_{23}$ to $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ | $A_9$ | $A_8$ | $A_7$ | $A_6$ | $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | $A_0$ |
| Row address | 0 | 0 | 8 bits | $A_{23}$ to $A_{13}$ | $A_{20}$ | $A_{19}$ | $A_{18}$ | $A_{17}$ | $A_{16}$ | $A_{15}$ | $A_{14}$ | $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ | $A_9$ | $A_8$ |
| | | 1 | 9 bits | $A_{23}$ to $A_{13}$ | $A_{12}$ | $A_{20}$ | $A_{19}$ | $A_{18}$ | $A_{17}$ | $A_{16}$ | $A_{15}$ | $A_{14}$ | $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ | $A_9$ |
| | 1 | 0 | 10 bits | $A_{23}$ to $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{20}$ | $A_{19}$ | $A_{18}$ | $A_{17}$ | $A_{16}$ | $A_{15}$ | $A_{14}$ | $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ |
| | | 1 | Setting prohibited | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Column address | — | — | — | $A_{23}$ to $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ | $A_9$ | $A_8$ | $A_7$ | $A_6$ | $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | $A_0$ |

### 6.5.4 Data Bus

If the bit in ABWCR corresponding to an area designated as DRAM space is set to 1, that area is designated as 8-bit DRAM space; if the bit is cleared to 0, the area is designated as 16-bit DRAM space. In 16-bit DRAM space, × 16-bit configuration DRAM can be connected directly.

In 8-bit DRAM space the upper half of the data bus, $D_{15}$ to $D_8$, is enabled, while in 16-bit DRAM space both the upper and lower halves of the data bus, $D_{15}$ to $D_0$, are enabled.

Access sizes and data alignment are the same as for the basic bus interface: see section 6.4.2, Data Size and Data Alignment.

### 6.5.5 Pins Used for DRAM Interface

Table 6-7 shows the pins used for DRAM interfacing and their functions.

**Table 6-7 DRAM Interface Pins**

| Pin | With DRAM Setting | Name | I/O | Function |
|---|---|---|---|---|
| $\overline{HWR}$ | $\overline{WE}$/$\overline{UWE}$ | Write enable/upper write enable | Output | When 2-CAS system is set, write enable for DRAM space access.<br>When 2-WE system is set, upper write enable for DRAM space access. |
| $\overline{LWR}$ | $\overline{LCAS}$/$\overline{LWE}$ | Lower column address strobe/lower write enable | Output | When 2-CAS system (LCASS = 1) is set, lower column address strobe for DRAM space access.<br>When 2-WE system is set, lower write enable for DRAM space access. |
| $\overline{LCAS}$ | $\overline{LCAS}$ | Lower column address strobe | Output | Lower column address strobe for access to 2-CAS type (LCASS = 0) DRAM space. |
| $\overline{CS}_2$ | $\overline{RAS}_2$ | Row address strobe 2 | Output | Row address strobe when area 2 is designated as DRAM space. |
| $\overline{CS}_3$ | $\overline{RAS}_3$ | Row address strobe 3 | Output | Row address strobe when area 3 is designated as DRAM space. |
| $\overline{CS}_4$ | $\overline{RAS}_4$ | Row address strobe 4 | Output | Row address strobe when area 4 is designated as DRAM space. |
| $\overline{CS}_5$ | $\overline{RAS}_5$ | Row address strobe 5 | Output | Row address strobe when area 5 is designated as DRAM space. |
| $\overline{CAS}$ | $\overline{CAS}$/$\overline{UCAS}$ | Column address strobe/ upper column address strobe | Output | When 2-WE system is set, column address strobe.<br>When 2-CAS system is set, upper column address strobe. |
| $\overline{WAIT}$ | $\overline{WAIT}$ | Wait | Input | Wait request signal |
| $A_{12}$ to $A_0$ | $A_{12}$ to $A_0$ | Address pins | Output | Row address/column address multiplexed output |
| $D_{15}$ to $D_0$ | $D_{15}$ to $D_0$ | Data pins | I/O | Data input/output pins |

### 6.5.6 Basic Timing

Figure 6-15 shows the basic access timing for DRAM space. The basic DRAM access timing is 4 states. Unlike the basic bus interface, the corresponding bits in ASTCR control only enabling or disabling of wait insertion, and do not affect the number of access states. When the corresponding bit in ASTCR is cleared to 0, wait states cannot be inserted in the DRAM access cycle.

The 4 states of the basic timing consist of one $T_p$ (precharge cycle) state, one $T_r$ (row address output cycle), and two $T_c$ (column address output cycle) states, $T_{c1}$ and $T_{c2}$.

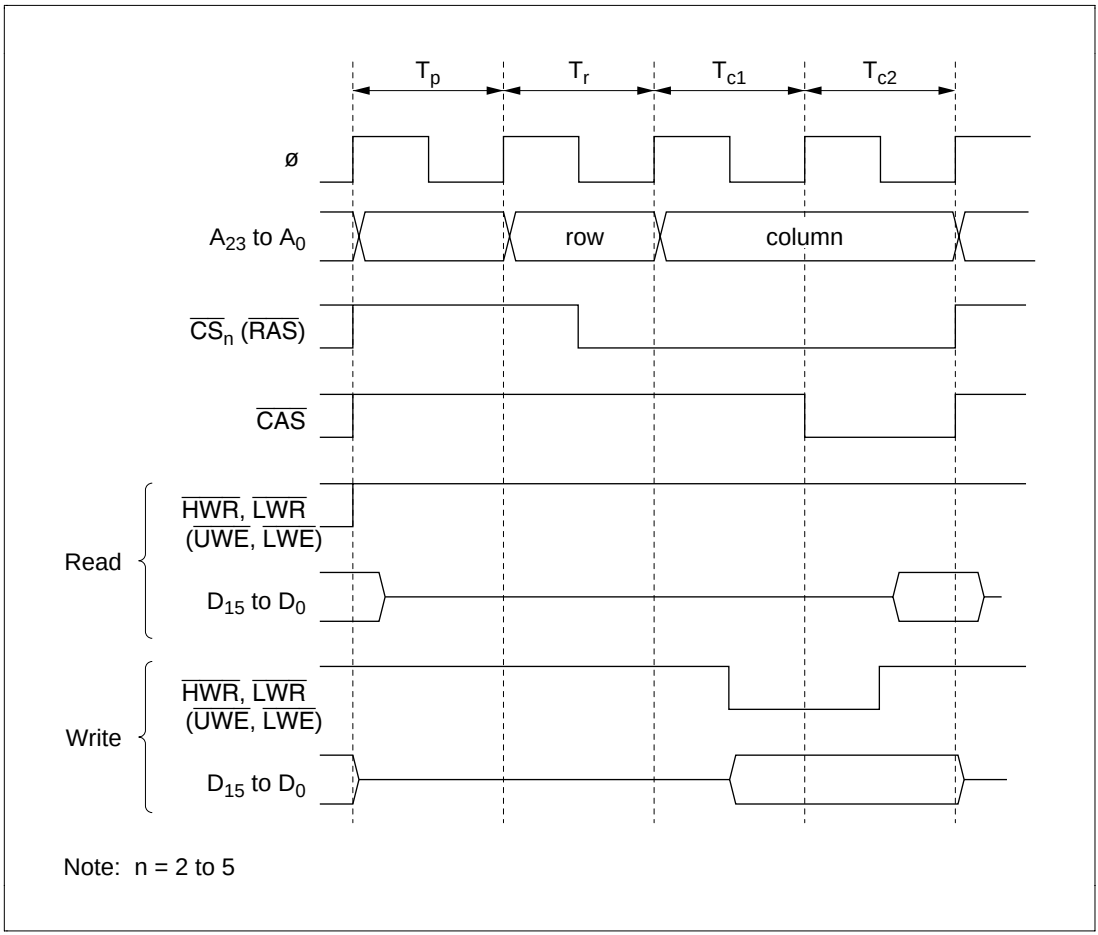



**Figure 6-15 Basic Access Timing (2-WE System)**

### 6.5.7 Precharge State Control

When DRAM is accessed, RAS precharging time must be secured. With the H8S/2655 Series, one $T_p$ state is always inserted when DRAM space is accessed. This can be changed to two $T_p$ states by setting the TPC bit in MCR to 1. Set the appropriate number of $T_p$ cycles according to the DRAM connected and the operating frequency of the H8S/2655 Series. Figure 6-16 shows the timing when two $T_p$ states are inserted.

When the TCP bit is set to 1, two $T_p$ states are also used for refresh cycles.

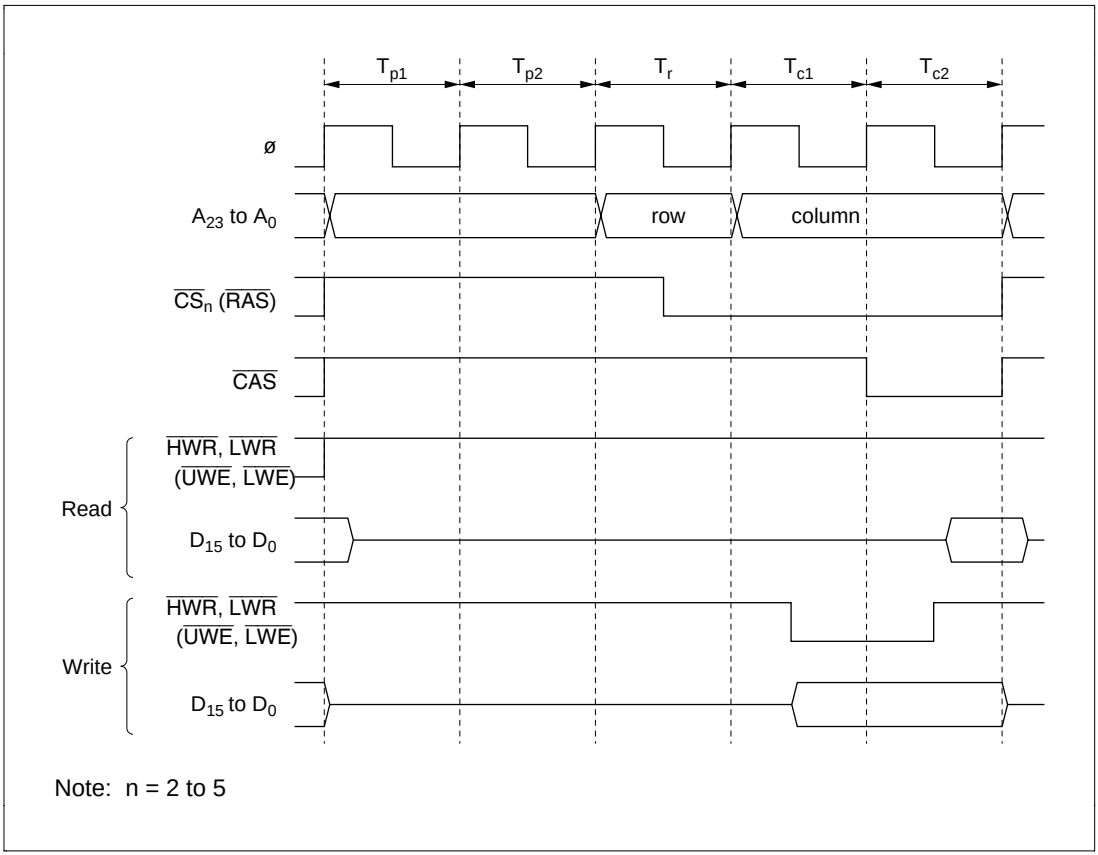



**Figure 6-16 Timing with Two Precharge States (2-WE System)**

### 6.5.8 Wait Control

There are two ways of inserting wait states in a DRAM access cycle: program wait insertion and pin wait insertion using the $\overline{\text{WAIT}}$ pin.

**Program Wait Insertion**

When the bit in ASTCR corresponding to an area designated as DRAM space is set to 1, from 0 to 3 wait states can be inserted automatically between the $T_{c1}$ state and $T_{c2}$ state, according to the settings of WCRH and WCRL.

**Pin Wait Insertion**

When the WAITE bit in BCRH is set to 1, wait input by means of the $\overline{\text{WAIT}}$ pin is enabled regardless of the setting of the AST bit in ASTCR. When DRAM space is accessed in this state, a program wait is first inserted. If the $\overline{\text{WAIT}}$ pin is low at the falling edge of ø in the last $T_{c1}$ or $T_w$ state, another $T_w$ state is inserted. If the $\overline{\text{WAIT}}$ pin is held low, $T_w$ states are inserted until it goes high.

Figure 6-17 shows an example of wait state insertion timing.

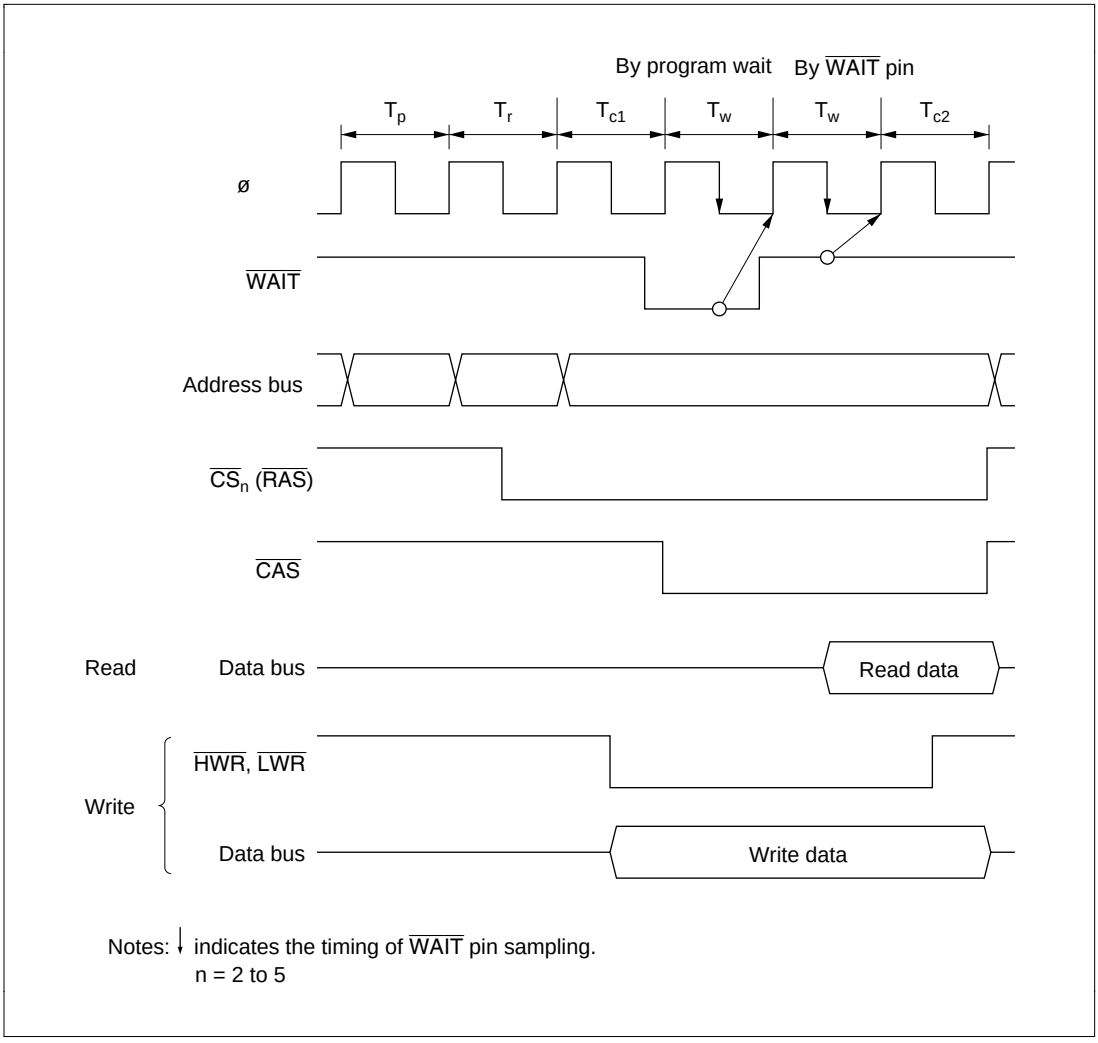



**Figure 6-17 Example of Wait State Insertion Timing**

### 6.5.9 Byte Access Control

When DRAM with a × 16 configuration is connected, the control signals needed for byte access differ depending on the kind of DRAM used. Either the 2-CAS system or the 2-WE system can be selected according to the setting of the CW2 bit in MCR.

**2-CAS System**

When the CW2 bit in MCR is cleared to 0, the 2-CAS system is selected. With this system, $\overline{\text{UCAS}}$, $\overline{\text{LCAS}}$, and $\overline{\text{WE}}$ signals are output. Use of the $\overline{\text{LWR}}$ pin or the $\overline{\text{LCAS}}$ pin for the $\overline{\text{LCAS}}$ signal can be selected by means of the LCASS bit.

**(a) When LCASS = 0**

Figure 6-18 (a) shows the control timing in the 2-CAS system (LCASS = 0), and figure 6-19 (a) shows an example of 2-CAS system (LCASS = 0) DRAM connection.

**(b) When LCASS = 1**

Figure 6-18 (b) shows the control timing in the 2-CAS system (LCASS = 1), and figure 6-19 (b) shows an example of 2-CAS system (LCASS = 1) DRAM connection.

In this case, since the $\overline{\text{LWR}}$ pin is used for the $\overline{\text{LCAS}}$ signal, RAS down mode cannot be used.

Regardless of the ICIS1 and ICIS0 bits, when non-DRAM space is accessed following a DRAM space access, an idle cycle (TDI) is inserted after the DRAM space access.

Access to another space is not performed during CBR refreshing; access to another space is performed after insertion of an idle cycle (TRI).

**2-WE System**

When the CW2 bit in MCR is set to 1, the 2-WE system is selected. With this system, $\overline{\text{CAS}}$, $\overline{\text{UWE}}$, and $\overline{\text{LWE}}$ signals are output. Figure 6-20 shows the control timing in the 2-WE system, and figure 6-21 shows an example of DRAM connection using this system.

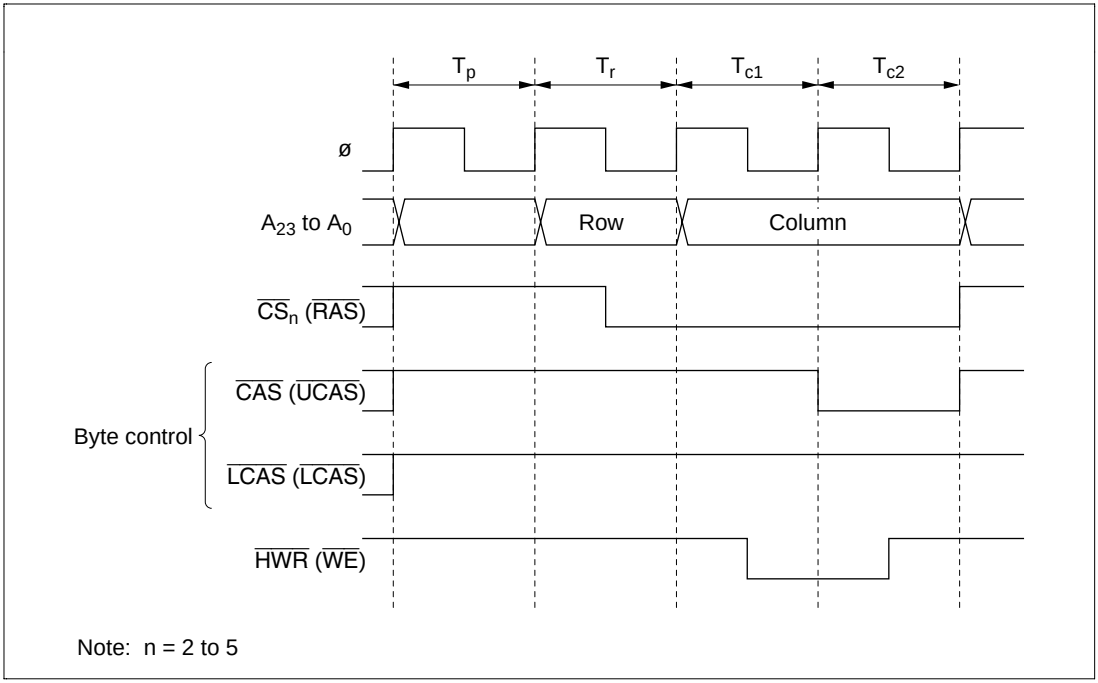



**Figure 6-18 (a) 2-CAS System (LCASS = 0) Control Timing (Upper Byte Write Access)**

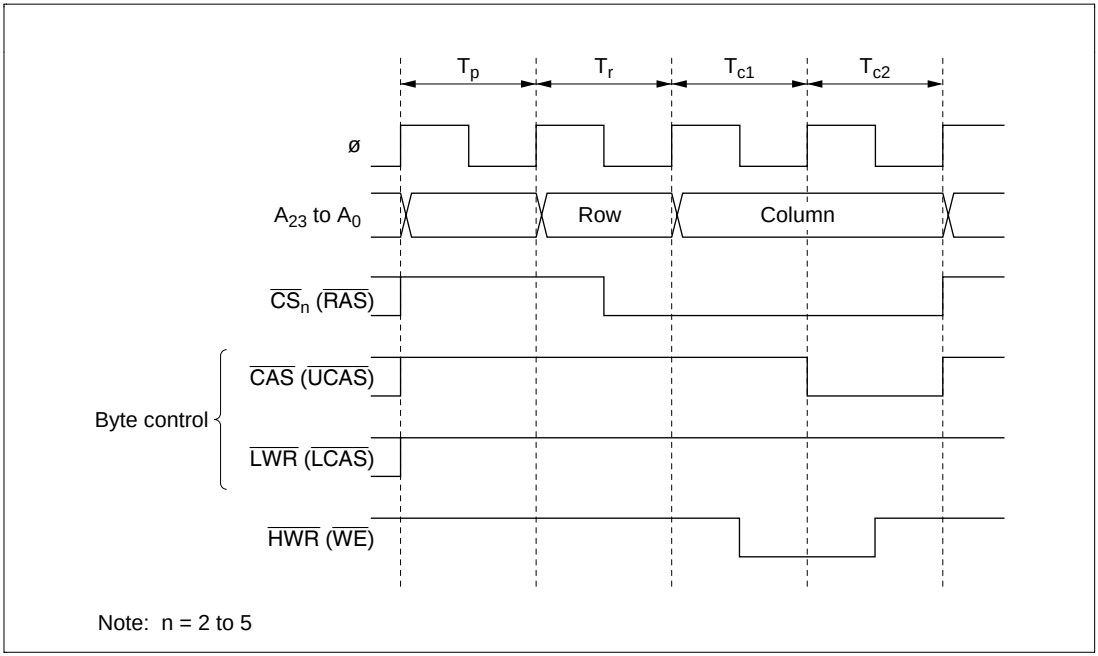



**Figure 6-18 (b) 2-CAS System (LCASS = 1) Control Timing (Upper Byte Write Access)**

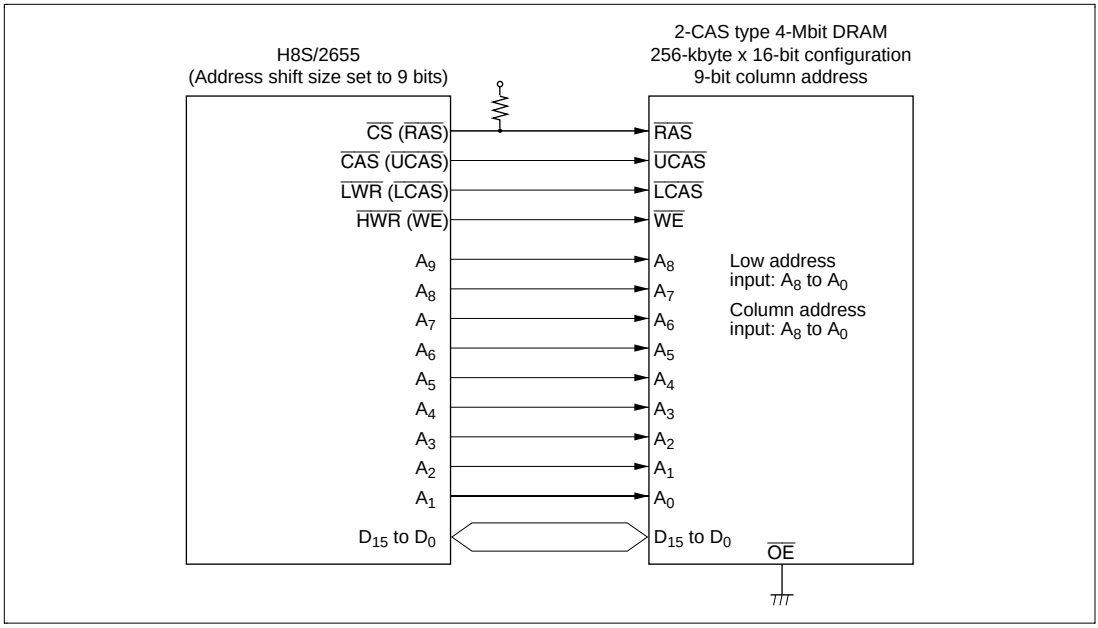



**Figure 6-19 (a) Example of 2-CAS System (LCASS = 0) DRAM Connection**

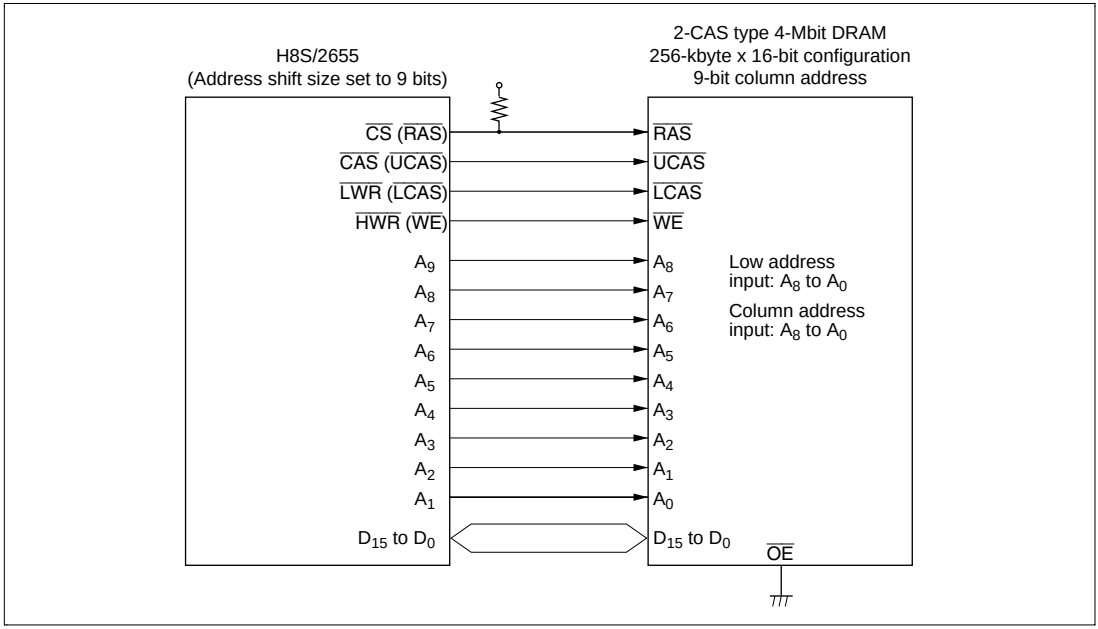



**Figure 6-19 (b) Example of 2-CAS System (LCASS = 1) DRAM Connection**

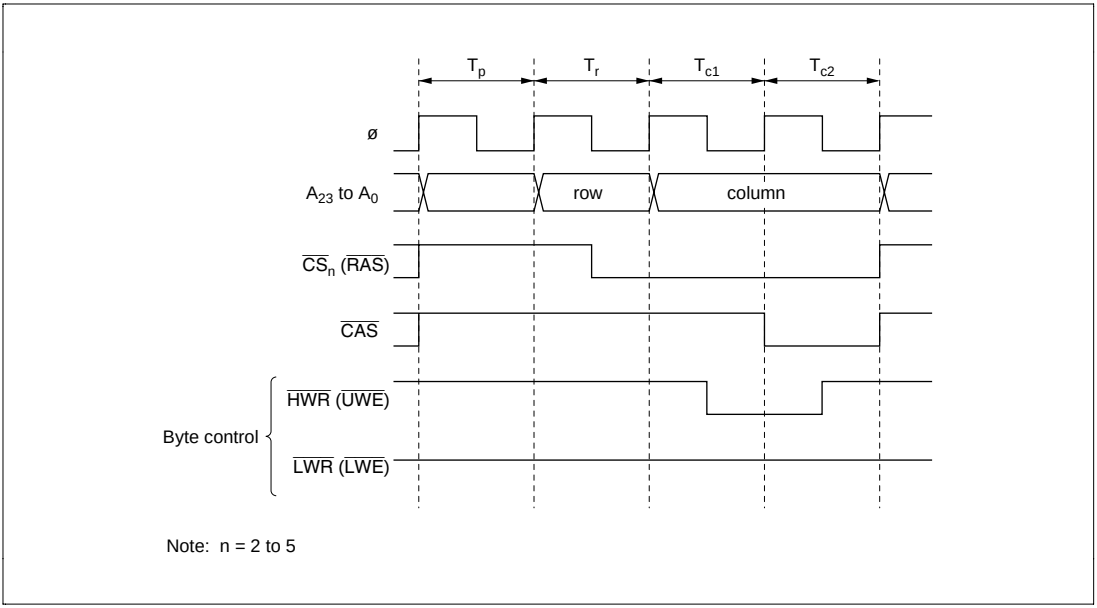



**Figure 6-20 2-WE System Control Timing (Upper Byte Access)**

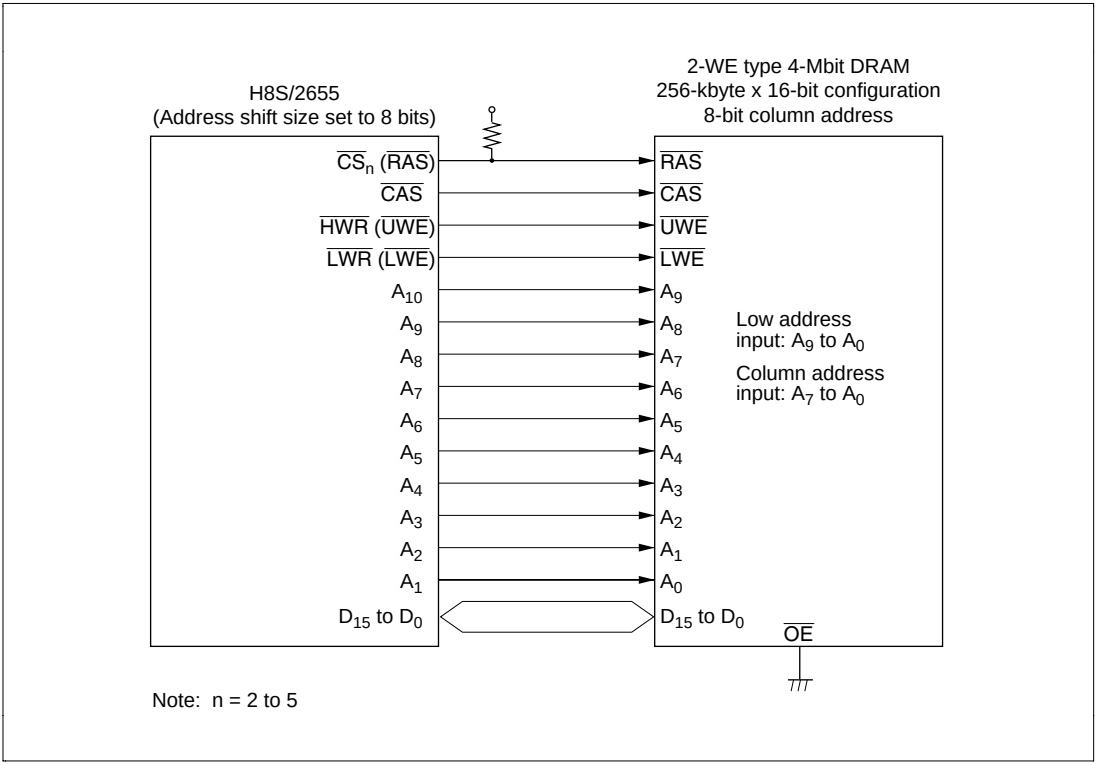



**Figure 6-21 Example of 2-WE Type DRAM Connection**

### 6.5.10 Burst Operation

With DRAM, in addition to full access (normal access) in which data is accessed by outputting a row address for each access, a fast page mode is also provided which can be used when making a number of consecutive accesses to the same row address. This mode enables fast (burst) access of data by simply changing the column address after the row address has been output. Burst access can be selected by setting the BE bit in MCR to 1.

#### (1) Burst Access (Fast Page Mode) Operation Timing

Figure 6-22 shows the operation timing for burst access. When there are consecutive access cycles for DRAM space, the $\overline{CAS}$ signal and column address output cycles (two states) continue as long as the row address is the same for consecutive access cycles. The row address used for the comparison is set with bits MXC1 and MXC0 in MCR.

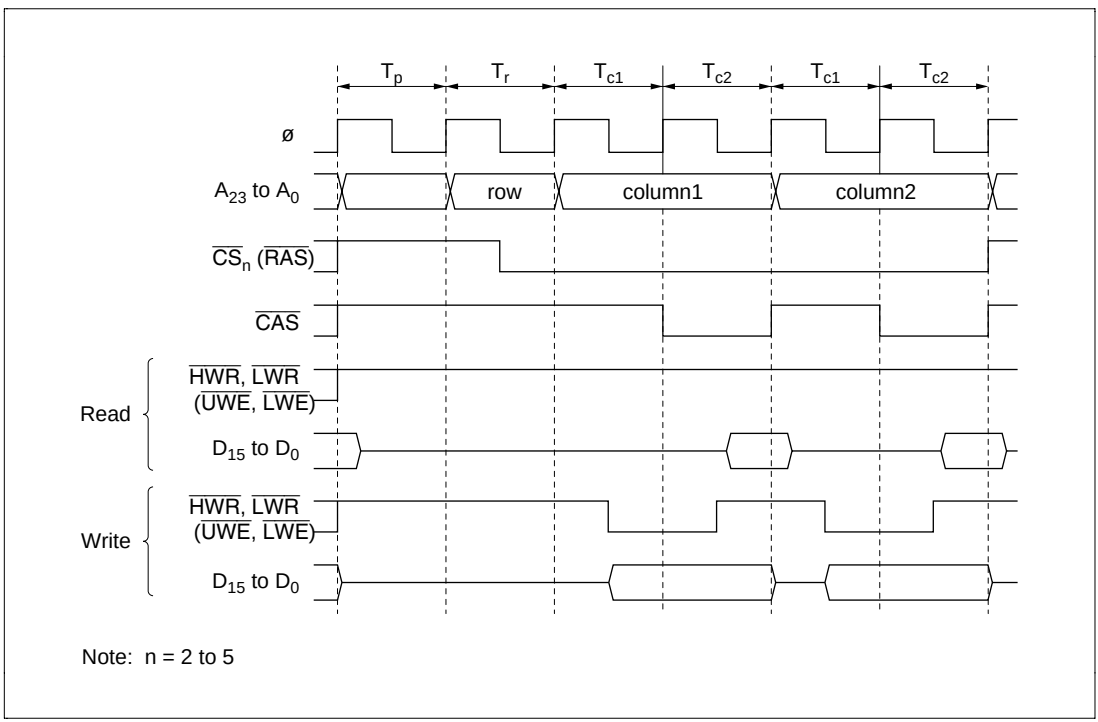



**Figure 6-22 Operation Timing in Fast Page Mode (2-WE System)**

The bus cycle can also be extended in burst access by inserting wait states. The wait state insertion method and timing are the same as for full access. For details, see section 6.5.8, Wait Control.

### (2) RAS Down Mode and RAS Up Mode

Even when burst operation is selected, it may happen that access to DRAM space is not continuous, but is interrupted by access to another space. In this case, if the $\overline{RAS}$ signal is held low during the access to the other space, burst operation can be resumed when the same row address in DRAM space is accessed again.

RAS down mode cannot be used with the 2-CAS system (when LCASS = 1).

- RAS down mode

  To select RAS down mode, set the RCDM bit in MCR to 1. If access to DRAM space is interrupted and another space is accessed, the $\overline{RAS}$ signal is held low during the access to the other space, and burst access is performed if the row address of the next DRAM space access is the same as the row address of the previous DRAM space access. Figure 6-23 shows an example of the timing in RAS down mode.

  Note, however, that the $\overline{RAS}$ signal will go high if a refresh operation interrupts RAS down mode.

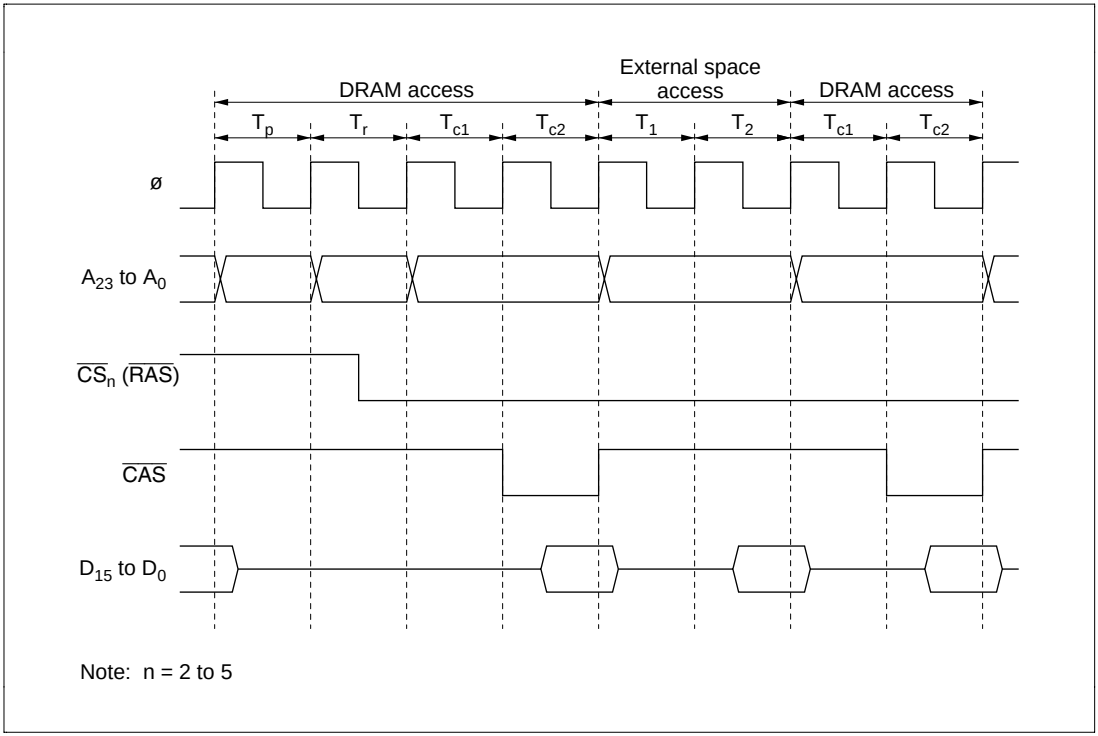



**Figure 6-23 Example of Operation Timing in RAS Down Mode**

- RAS up mode

  To select RAS up mode, clear the RCDM bit in MCR to 0. Each time access to DRAM space is interrupted and another space is accessed, the $\overline{RAS}$ signal goes high again. Burst operation is only performed if DRAM space is continuous. Figure 6-24 shows an example of the timing in RAS up mode.

  In the case of burst ROM space access, the $\overline{RAS}$ signal is not restored to the high level.

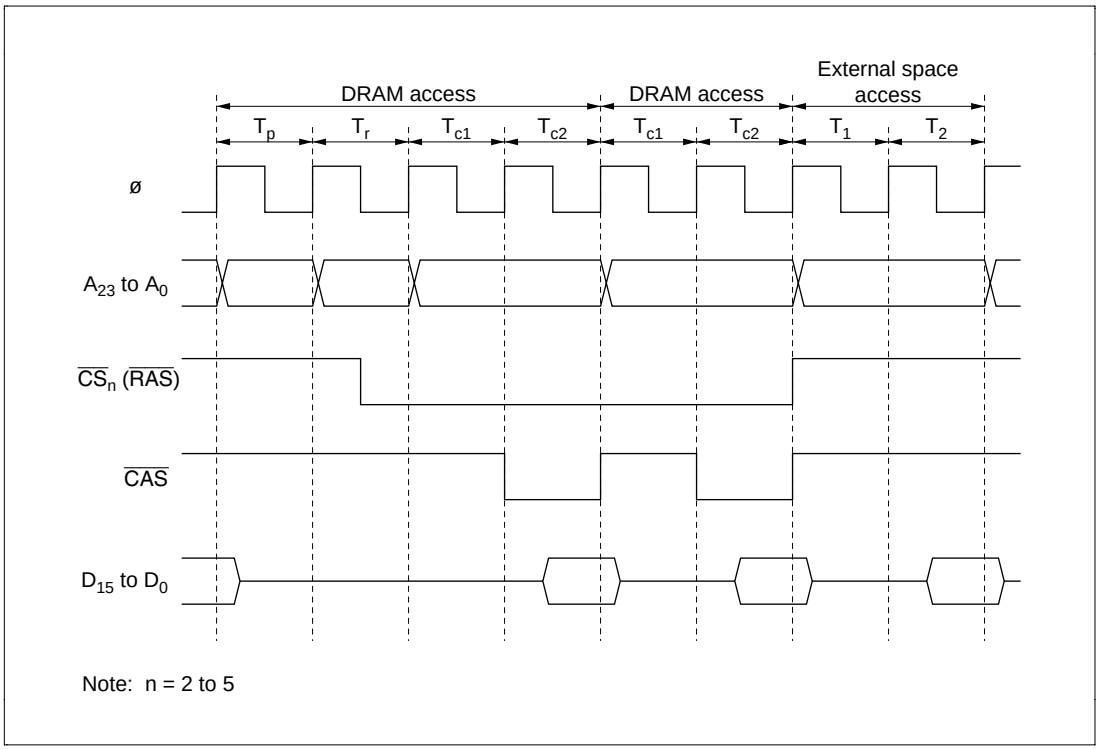



**Figure 6-24 Example of Operation Timing in RAS Up Mode**

### 6.5.11 Caution Concerning 2-CAS System

When the 2-CAS system (LCASS = 1) is selected, if normal (non-DRAM) space is accessed following a DRAM space access, an idle cycle ($T_{DI}$) is inserted after the DRAM space access. An idle cycle is not inserted if the DRAM space access is followed by an on-chip memory access or DRAM space access.

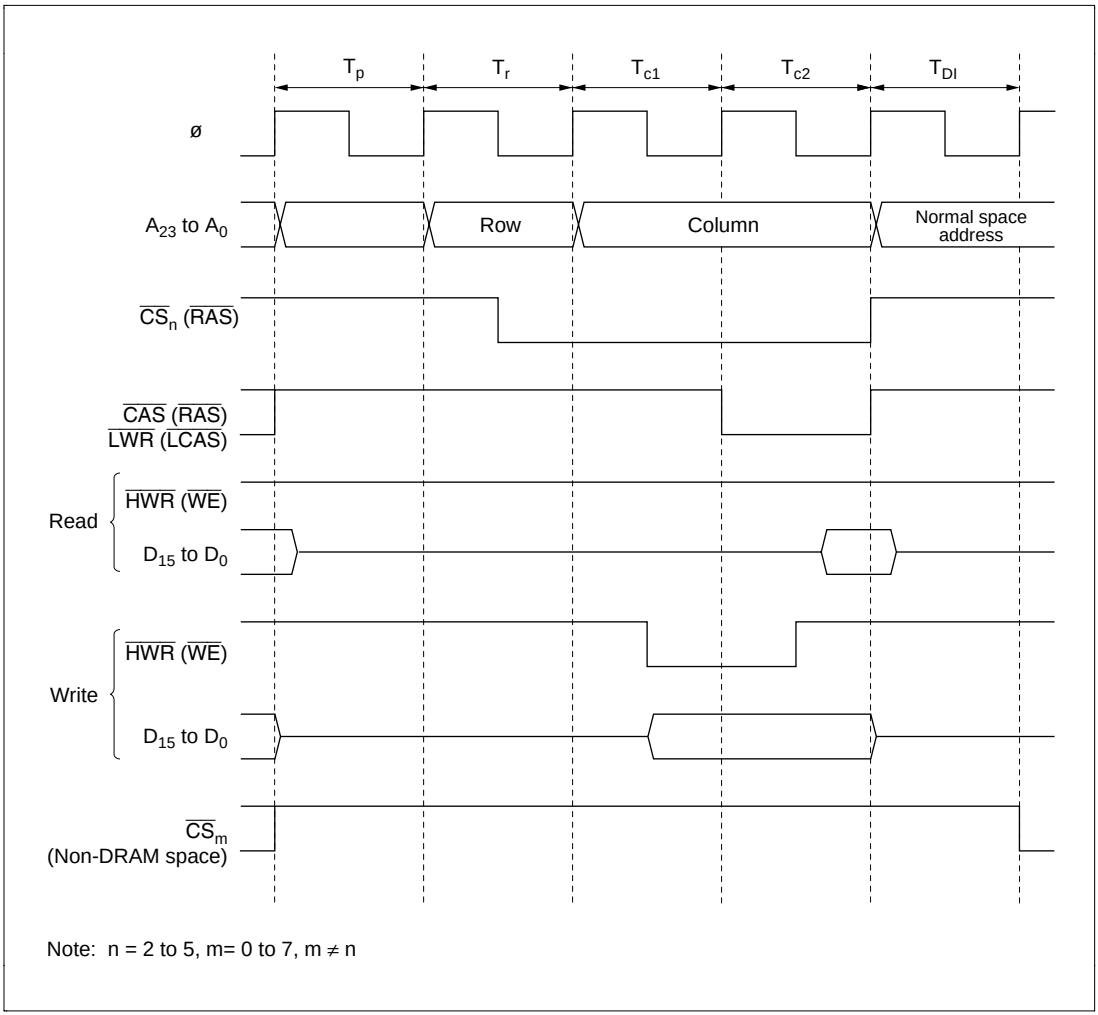



**Figure 6-25 Idle Cycle Insertion with 2-CAS System (LCASS = 1)**

### 6.5.12 Refresh Control

The H8S/2655 Series is provided with a DRAM refresh control function. Either of two refreshing methods can be selected: CAS-before-RAS (CBR) refreshing, or self-refreshing.

**(1) CAS-before-RAS (CBR) Refreshing**

To select CBR refreshing, set the RFSHE bit in DRAMCR to 1, and clear the RMODE bit to 0.

With CBR refreshing, RTCNT counts up using the input clock selected by bits CKS2 to CKS0 in DRAMCR, and when the count matches the value set in RTCOR (compare match), refresh control is performed. At the same time, RTCNT is reset and starts counting again from H'00. Refreshing is thus repeated at fixed intervals determined by RTCOR and bits CKS2 to CKS0. Set a value in RTCOR and bits CKS2 to CKS0 that will meet the refreshing interval specification for the DRAM used.

When bits CKS2 to CKS0 are set, RTCNT starts counting up. RTCNT and RTCOR settings should therefore be completed before setting bits CKS2 to CKS0.

Do not clear the CMF flag when refresh control is being performed (RFSHE = 1).

RTCNT operation is shown in figure 6-26, compare match timing in figure 6-27, and 2-WE and 2-CAS system CBR refresh timings in figures 6-28 and 29.

An access to another normal space is performed during the 2-WE system or 2-CAS system (LCASS = 0) refresh period. An access to another normal space is not performed during the 2-CAS system (LCASS = 1) refresh period, but following insertion of an idle cycle ($T_{RI}$) after refreshing is completed. An idle cycle ($T_{RI}$) is not inserted when an on-chip memory access or DRAM space access follows.

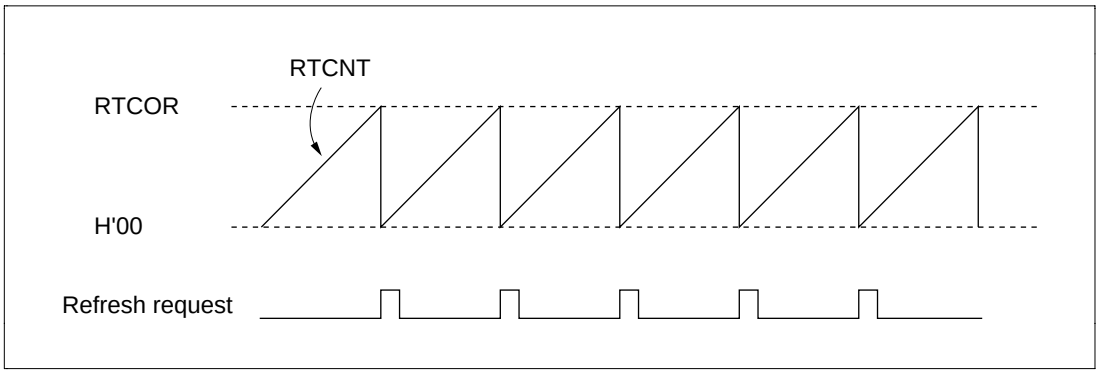



**Figure 6-26 RTCNT Operation**

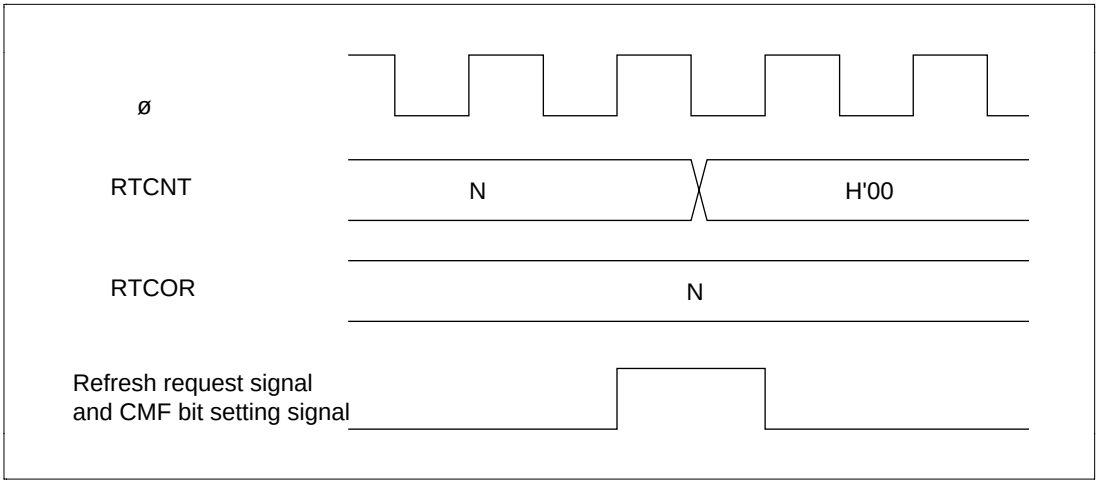



**Figure 6-27 Compare Match Timing**

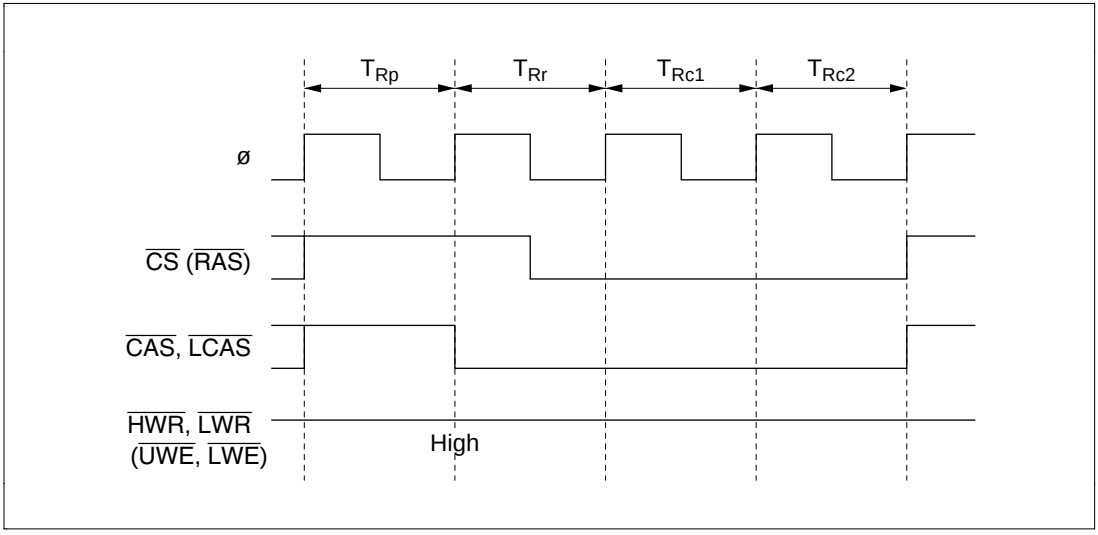



**Figure 6-28 2-WE System or 2-CAS System (LCASS = 0) CBR Refresh Timing (When RCW = 0 and CW2 = 1; or RCW = 0, CWZ = 0, and LCASS = 0)**

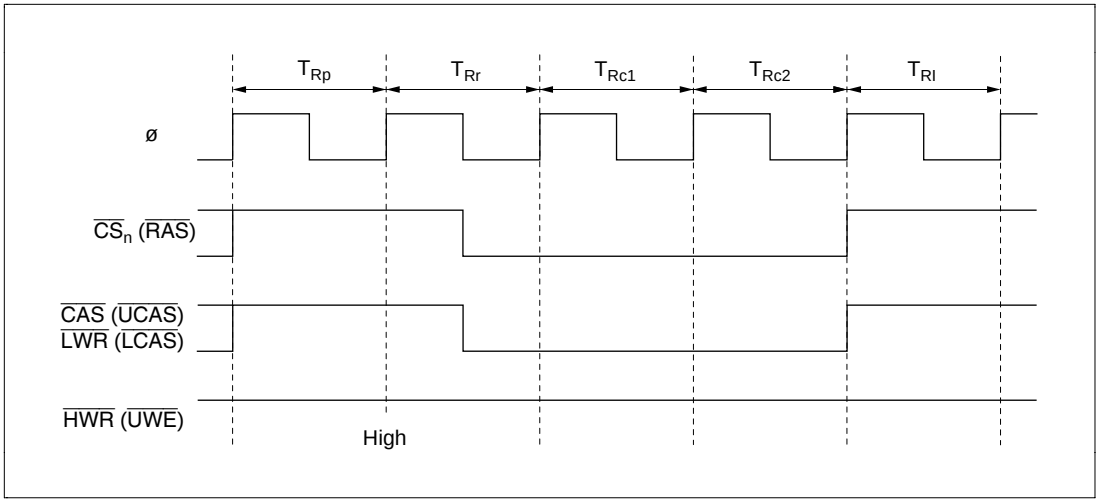



**Figure 6-29 2-CAS System (LCASS = 1) CBR Refresh Timing (When RCW = 0 and CW2 = 0, LCASS = 1)**

When the RCW bit is set to 1, $\overline{RAS}$ signal output is delayed by one cycle. The width of the $\overline{RAS}$ signal should be adjusted with bits RLW1 and RLW0. These bits are only enabled in refresh operations.

Figure 6-30 shows the timing when the RCW bit is set to 1.

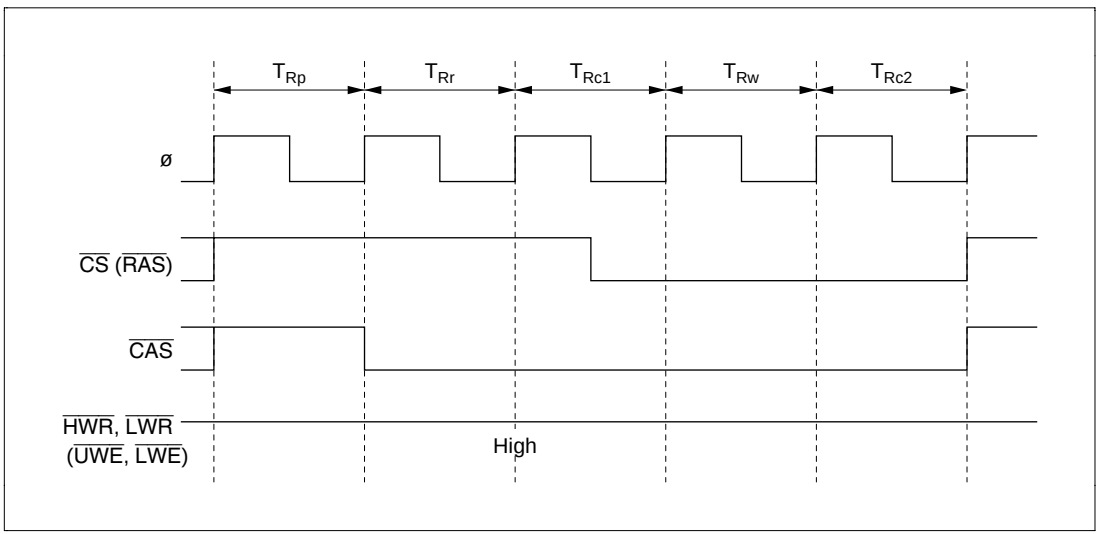



**Figure 6-30 CBR Refresh Timing (When RCW = 1, RLW1 = 0, RLW0 = 1, CW2 = 1)**

**(2) Self-Refreshing**

A self-refresh mode (battery backup mode) is provided for DRAM as a kind of standby mode. In this mode, refresh timing and refresh addresses are generated within the DRAM.

To select self-refreshing, set the RFSHE bit and RMODE bit in DRAMCR to 1. Then, when a SLEEP instruction is executed to enter software standby mode, the $\overline{CAS}$ and $\overline{RAS}$ signals are output and DRAM enters self-refresh mode, as shown in figures 6-31 (a) and (b).

When software standby mode is exited, the RMODE bit is cleared to 0 and self-refresh mode is cleared.

When switching to software standby mode, if there is a CBR refresh request, CBR refreshing is executed before self-refresh mode is entered.

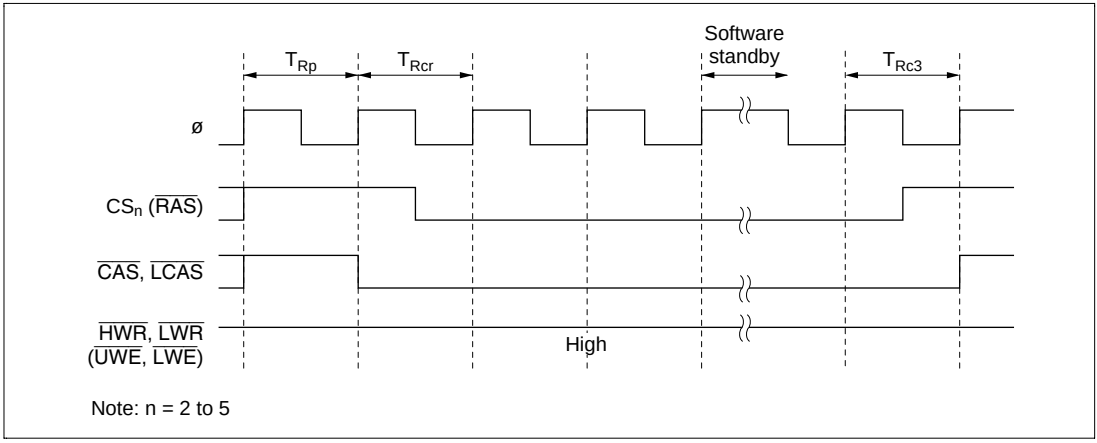



**Figure 6-31 (a) Self-Refresh Timing (When CW2 = 1, or CWZ = 1 and LCASS = 0)**

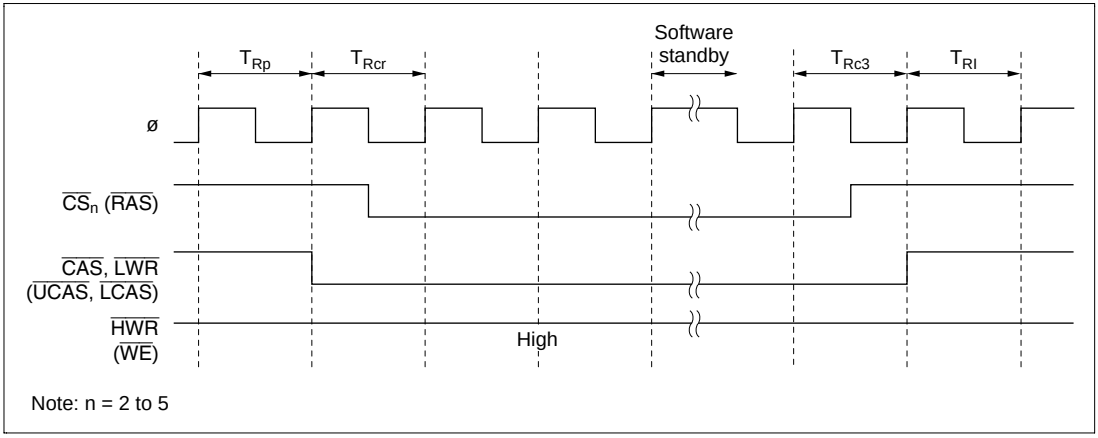



**Figure 6-31 (b) Self-Refresh Timing (When CW2 = 0, LCASS = 1)**

## 6.6 Pseudo-SRAM Interface

### 6.6.1 Overview

When the H8S/2655 Series is in advanced mode, external space areas 2 to 5 can be designated as pseudo-SRAM (PSRAM) space, and PSRAM interfacing performed. With the PSRAM interface, PSRAM can be directly connected to the H8S/2655 Series. A PSRAM space of 2, 4, or 8 Mbytes, or 128, 256, or 512 kbytes can be set by means of bits RMTS3 to RMTS0 in BCRH.

In the directly connected PSRAM, the refresh signal ($\overline{\text{RFSH}}$) and output enable signal ($\overline{\text{OE}}$) are multiplexed. Burst operation is also possible, using static column mode.

### 6.6.2 Setting PSRAM Space

Areas 2 to 5 are designated as PSRAM space by setting bits RMTS2 to RMTS0 in BCRH. The relation between the settings of bits RMTS2 to RMTS0 and PSRAM space is shown in table 6-8. Possible PSRAM space settings are: one area (area 2), two areas (areas 2 and 3), and four areas (areas 2 to 5).

**Table 6-8 Settings of Bits RMTS2 to RMTS0 and Corresponding PSRAM Spaces**

| RMTS2 | RMTS1 | RMTS0 | Area 5 | Area 4 | Area 3 | Area 2 |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | Normal space | | | PSRAM space |
| | 1 | 0 | Normal space | | PSRAM space | |
| | | 1 | PSRAM space | | | |

### 6.6.3 Data Bus

If the bit in ABWCR corresponding to an area designated as PSRAM space is set to 1, that area is designated as 8-bit PSRAM space; if the bit is cleared to 0, the area is designated as 16-bit PSRAM space.

In 8-bit PSRAM space the upper half of the data bus, $D_{15}$ to $D_8$, is enabled, while in 16-bit PSRAM space both the upper and lower halves of the data bus, $D_{15}$ to $D_0$, are enabled.

Access sizes and data alignment are the same as for the basic bus interface: see section 6.4.2, Data Size and Data Alignment.

### 6.6.4 Pins Used for PSRAM Interface

Table 6-9 shows the pins used for PSRAM interfacing and their functions.

**Table 6-9 PSRAM Interface Pins**

| Pin | With PSRAM Setting | Name | I/O | Function |
|---|---|---|---|---|
| $\overline{HWR}$ | $\overline{WE}$/$\overline{UWE}$ | Write enable/upper write enable | Output | Upper write enable when PSRAM space is designated for 16-bit access, or write enable when designated for 8-bit access. |
| $\overline{LWR}$ | $\overline{LWE}$ | Lower write enable | Output | Lower write enable when PSRAM space is designated for 16-bit access. |
| $\overline{CS}_2$ | $\overline{CE}_2$ | Chip enable 2 | Output | Chip enable signal when area 2 is designated as PSRAM space. |
| $\overline{CS}_3$ | $\overline{CE}_3$ | Chip enable 3 | Output | Chip enable signal when area 3 is designated as PSRAM space. |
| $\overline{CS}_4$ | $\overline{CE}_4$ | Chip enable 4 | Output | Chip enable signal when area 4 is designated as PSRAM space. |
| $\overline{CS}_5$ | $\overline{CE}_5$ | Chip enable 5 | Output | Chip enable signal when area 5 is designated as PSRAM space. |
| $\overline{CAS}$ | $\overline{OE}$/$\overline{RFSH}$ | Output enable/refresh | Output | Connected to PSRAM output enable/refresh dual-function pin |
| $\overline{WAIT}$ | $\overline{WAIT}$ | Wait | Input | Wait request signal input pin |
| $A_{20}$ to $A_0$ | $A_{20}$ to $A_0$ | Address pins | Output | Address output pins |
| $D_{15}$ to $D_0$ | $D_{15}$ to $D_0$ | Data pins | I/O | Data input/output pins |

### 6.6.5 Basic Timing

Figure 6-32 shows the basic access timing for PSRAM space. The basic PSRAM access timing is 4 states, including one $T_p$ (precharge cycle) state. Unlike the basic bus interface, the corresponding bits in ASTCR control only enabling or disabling of wait insertion, and do not affect the number of access states. When the corresponding bit in ASTCR is cleared to 0, wait states cannot be inserted in the PSRAM access cycle.

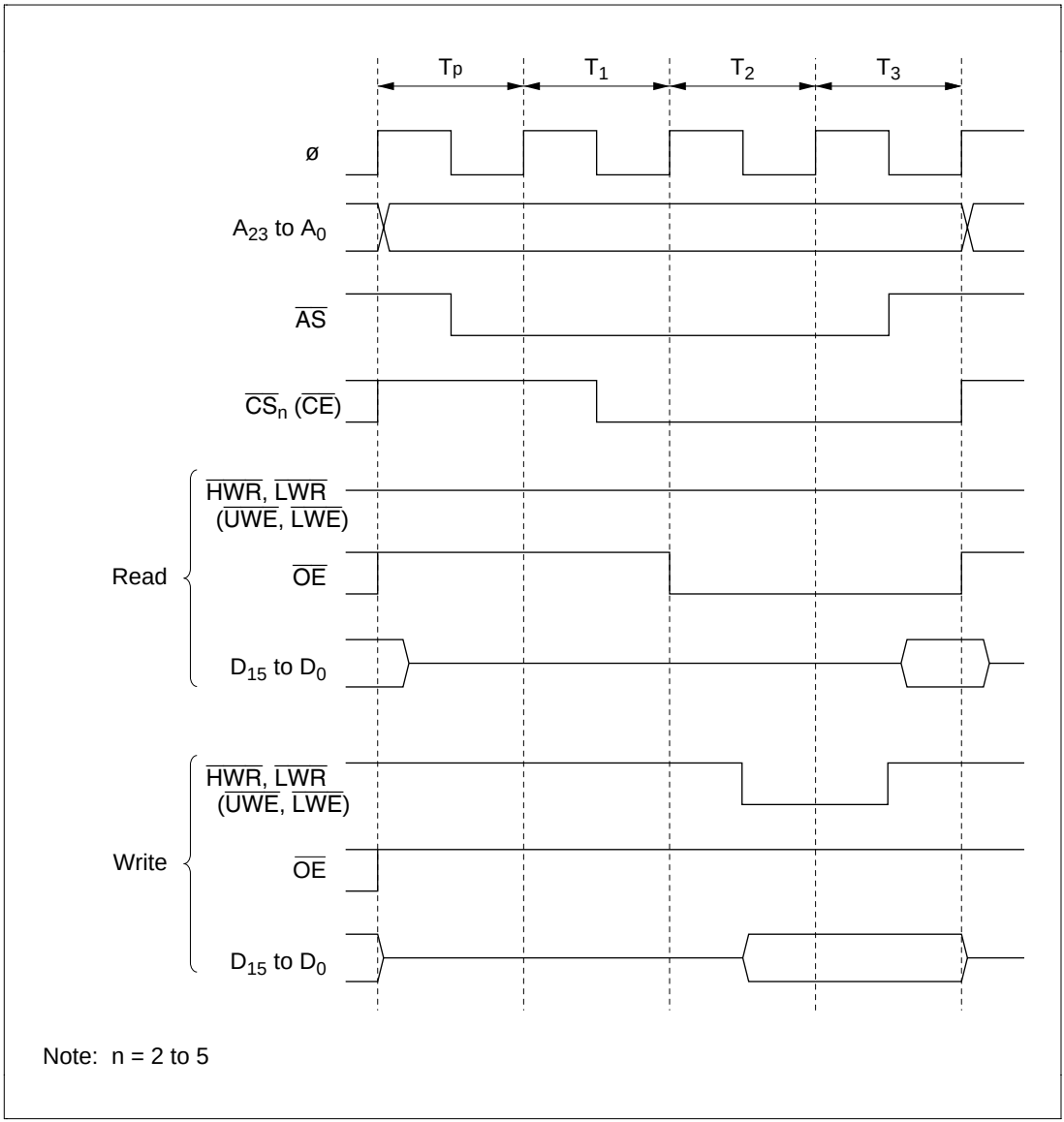



**Figure 6-32 Basic Access Timing**

### 6.6.6 Precharge State Control

When PSRAM is accessed, CE precharging time must be secured. With the H8S/2655 Series, one $T_p$ state is always inserted when PSRAM space is accessed. This can be changed to two $T_p$ states by setting the TPC bit in MCR to 1. Set the appropriate number of $T_p$ cycles according to the PSRAM connected and the operating frequency of the H8S/2655 Series. Figure 6-33 shows the timing when two $T_p$ states are inserted.

When the TCP bit is set to 1, two $T_p$ states are also used for refresh cycles.

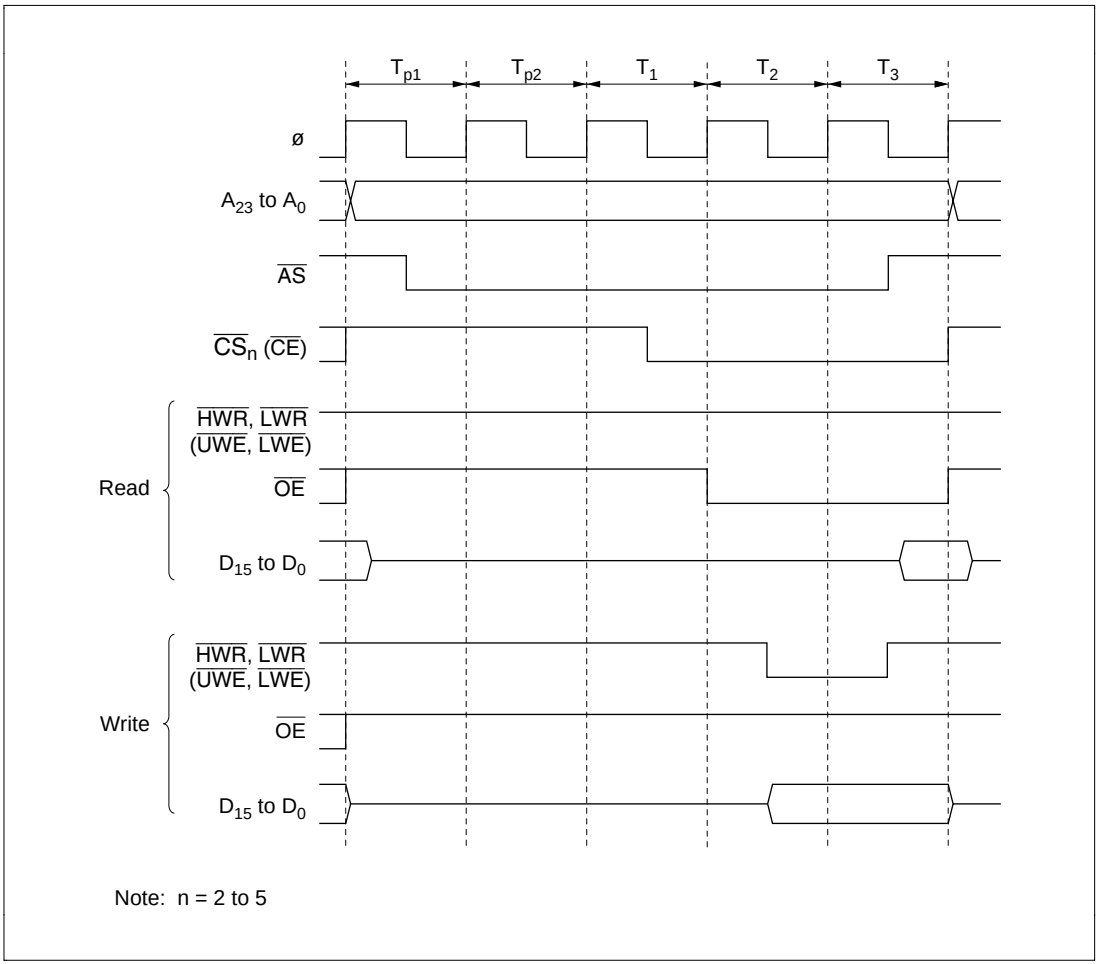



**Figure 6-33 Timing with Two Precharge States**

### 6.6.7 Wait Control

There are two ways of inserting wait states in a PSRAM access cycle: program wait insertion and pin wait insertion using the $\overline{\text{WAIT}}$ pin.

**(1) Program Wait Insertion**

When the bit in ASTCR corresponding to an area designated as PSRAM space is set to 1, from 0 to 3 wait states can be inserted automatically between the $T_2$ state and $T_3$ state, according to the settings of WCRH and WCRL.

**(2) Pin Wait Insertion**

When the WAITE bit in BCRL is set to 1, wait input by means of the $\overline{\text{WAIT}}$ pin is enabled regardless of the setting of the AST bit in ASTCR. When PSRAM space is accessed in this state, a program wait is first inserted. If the $\overline{\text{WAIT}}$ pin is low at the falling edge of ø in the last $T_2$ or $T_w$ state, another $T_w$ state is inserted. If the $\overline{\text{WAIT}}$ pin is held low, $T_w$ states are inserted until it goes high.

Figure 6-34 shows an example of wait state insertion timing.

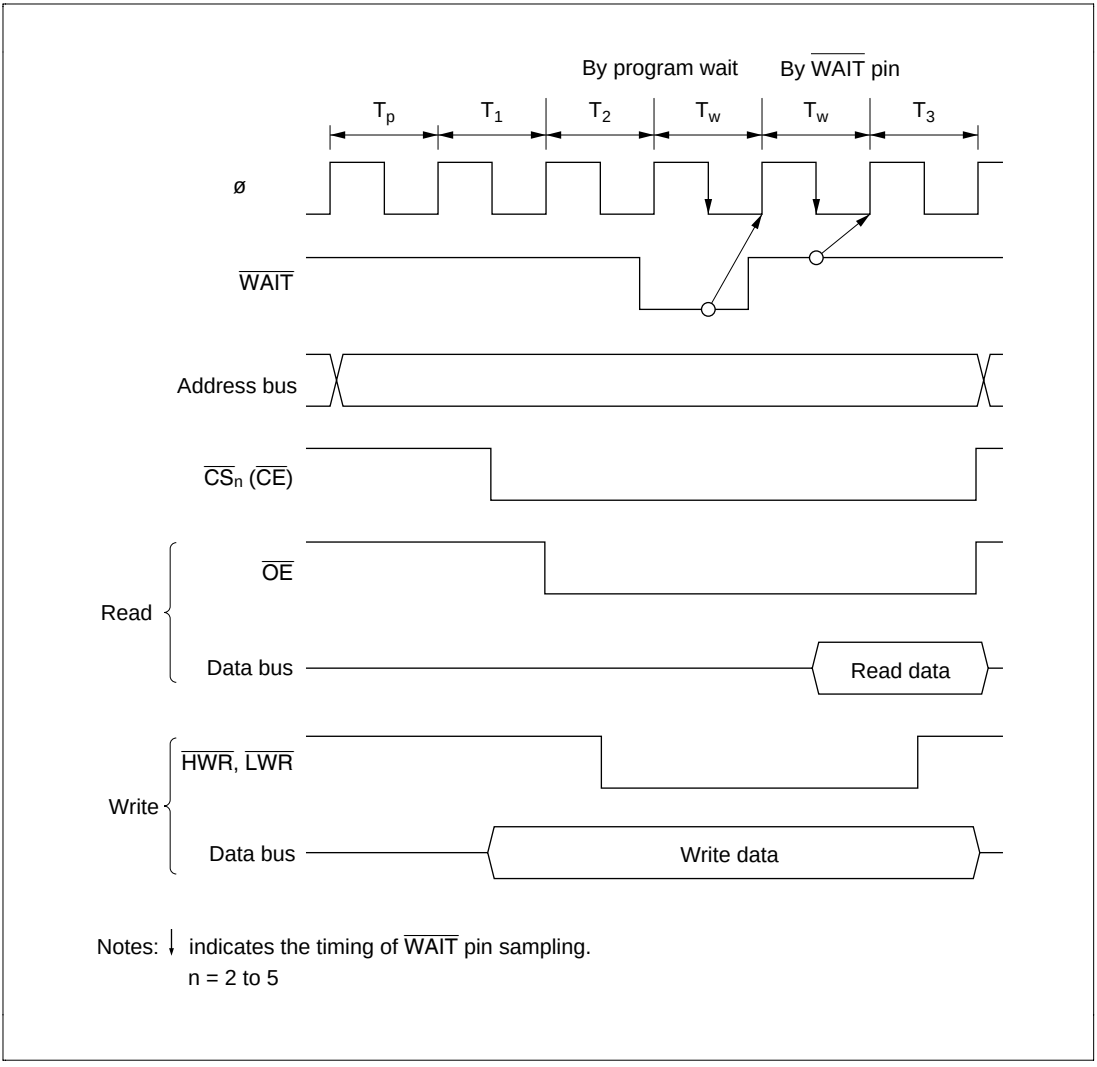



**Figure 6-34 Example of Wait State Insertion Timing**

### 6.6.8 Burst Operation

With PSRAM, a static column mode is provided which can be used when making a number of consecutive accesses to the same row address. This mode enables fast (burst) access of data by simply changing the column address while holding the $\overline{CS}$ signal low. Burst access can be selected by setting the BE bit in MCR to 1. PSRAM burst operation and burst ROM interface cannot be set at the same time.

Figure 6-35 shows the operation timing for burst access. When there are consecutive access cycles for PSRAM space, column address output cycles (two states) continue as long as the row address is the same for consecutive access cycles. The row address used for the comparison is set with bits MXC1 and MXC0 in MCR.

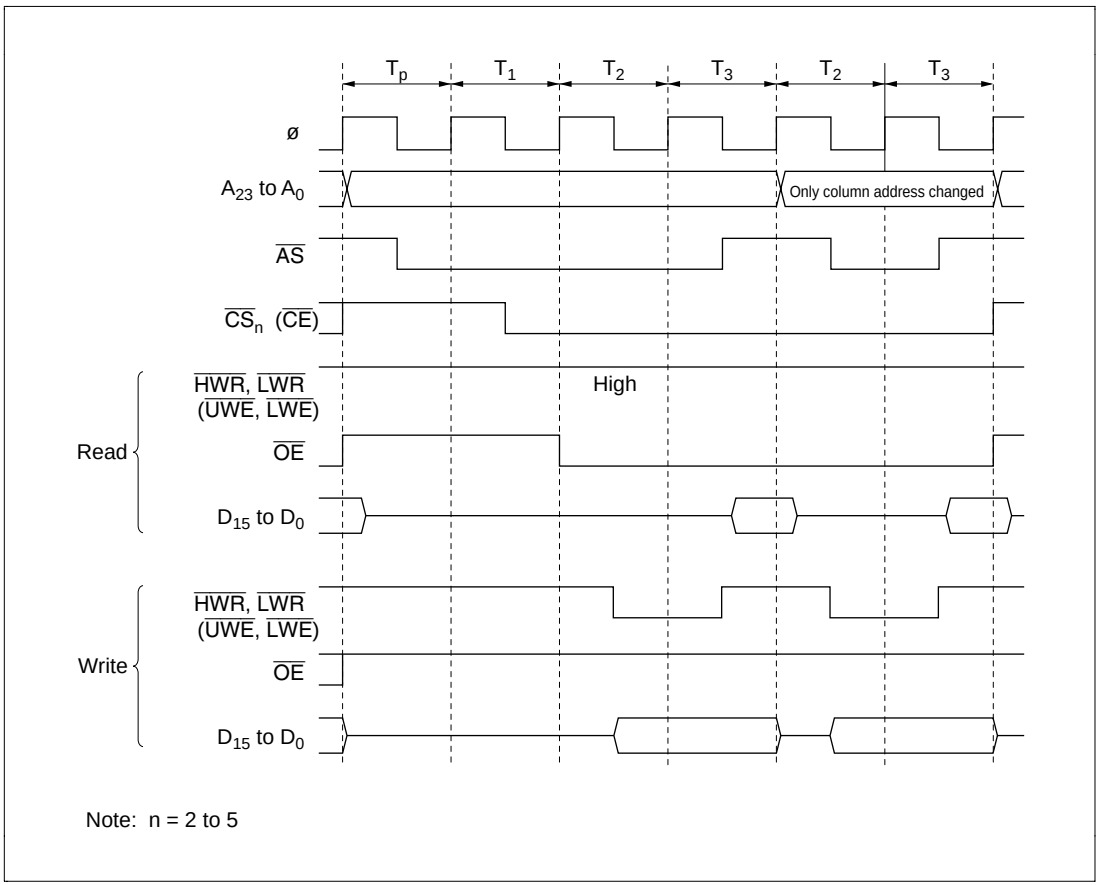



**Figure 6-35 Operation Timing for Burst Access**

The bus cycle can also be extended in burst access by inserting wait states. The wait state insertion method and timing are the same as for normal mode. For details, see section 6.6.7, Wait Control.

Even when burst operation is selected, it may happen that access to PSRAM space is not continuous, but is interrupted by access to another space. If access to PSRAM space is interrupted and on-chip memory or an internal I/O register is accessed, the $\overline{CE}$ signal is held low during that access, and burst access is performed if the row address of the next PSRAM space address is the same as the row address of the previous PSRAM space access. Figure 6-36 shows an example of the timing in burst access.

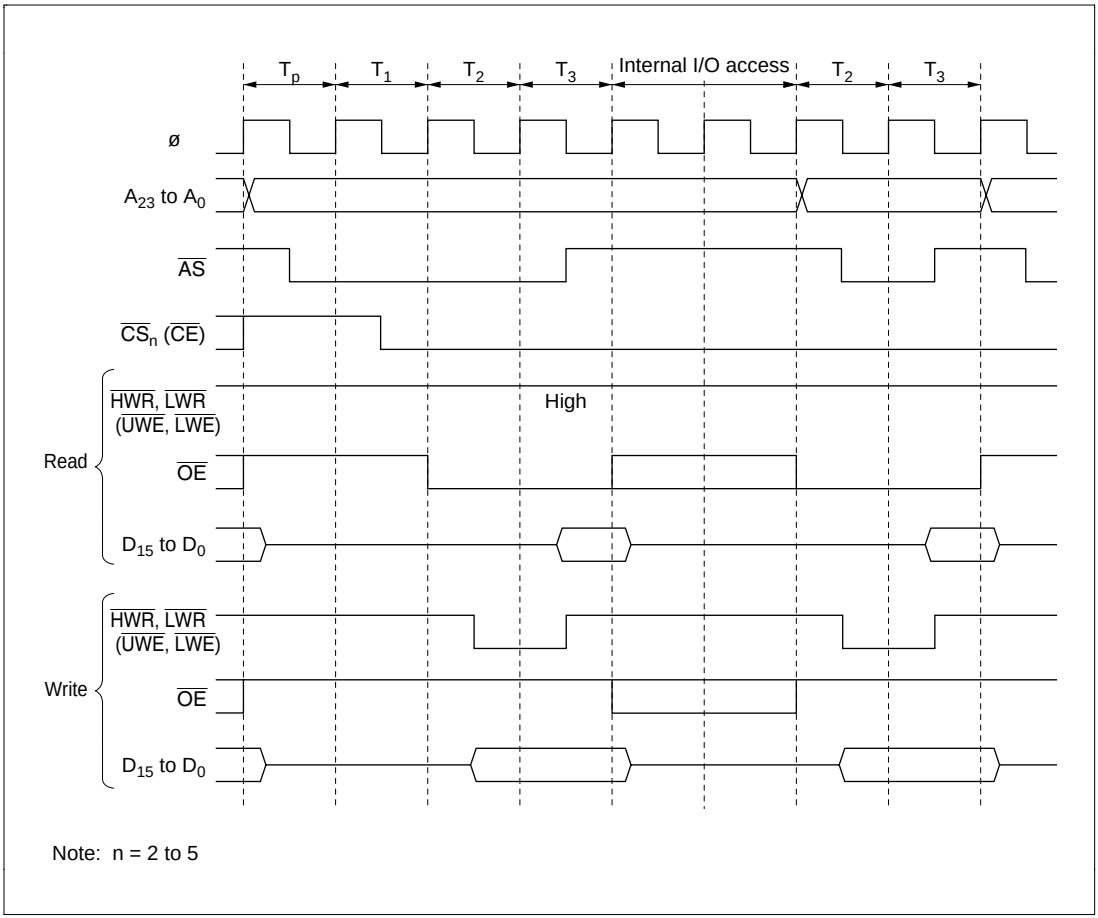



**Figure 6-36 Example of Operation Timing in Burst Access**

When access to PSRAM space is interrupted and another space is accessed, the $\overline{CE}$ signal goes high again. Burst operation is only performed if PSRAM space is continuous. Figure 6-37 shows an example of the timing.

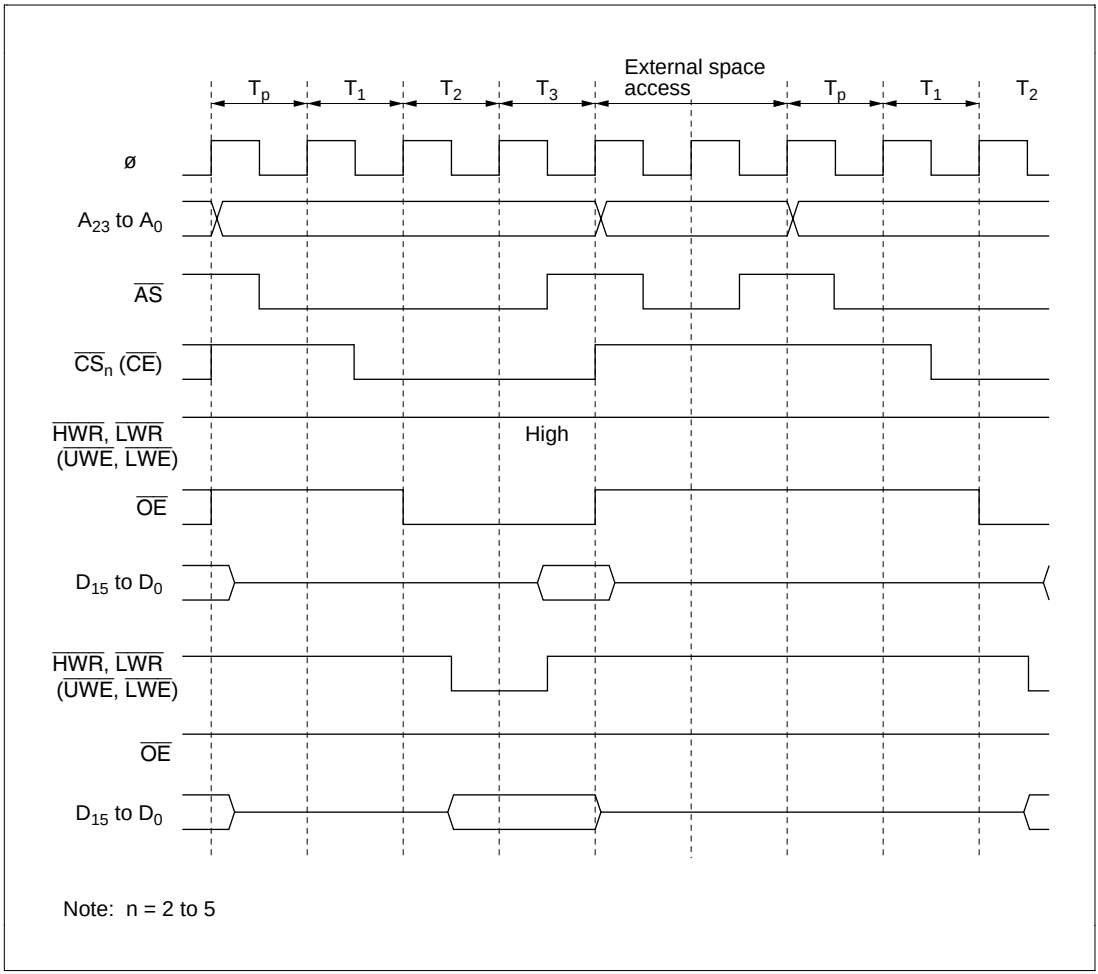



**Figure 6-37 Example of Interrupted Operation Timing in Burst Access**

### 6.6.9 Refresh Control

The H8S/2655 Series is provided with a PSRAM refresh control function. Either of two refreshing methods can be selected: auto-refreshing, or self-refreshing.

**(1) Auto-Refreshing**

To select auto-refreshing, set the RFSHE bit in DRAMCR to 1, and clear the RMODE bit to 0.

With auto-refreshing, RTCNT counts up using the input clock selected by bits CKS2 to CKS0 in DRAMCR, and when the count matches the value set in RTCOR (compare match), refresh control is performed. At the same time, RTCNT is reset and starts counting again from H'00. Refreshing is thus repeated at fixed intervals determined by RTCOR and bits CKS2 to CKS0. Set a value in RTCOR and bits CKS2 to CKS0 that will meet the refreshing interval specification for the PSRAM used.

When bits CKS2 to CKS0 are set, RTCNT starts counting up. RTCNT and RTCOR settings should therefore be completed before setting bits CKS2 to CKS0.

Auto-refresh timing is shown in figure 6-38.

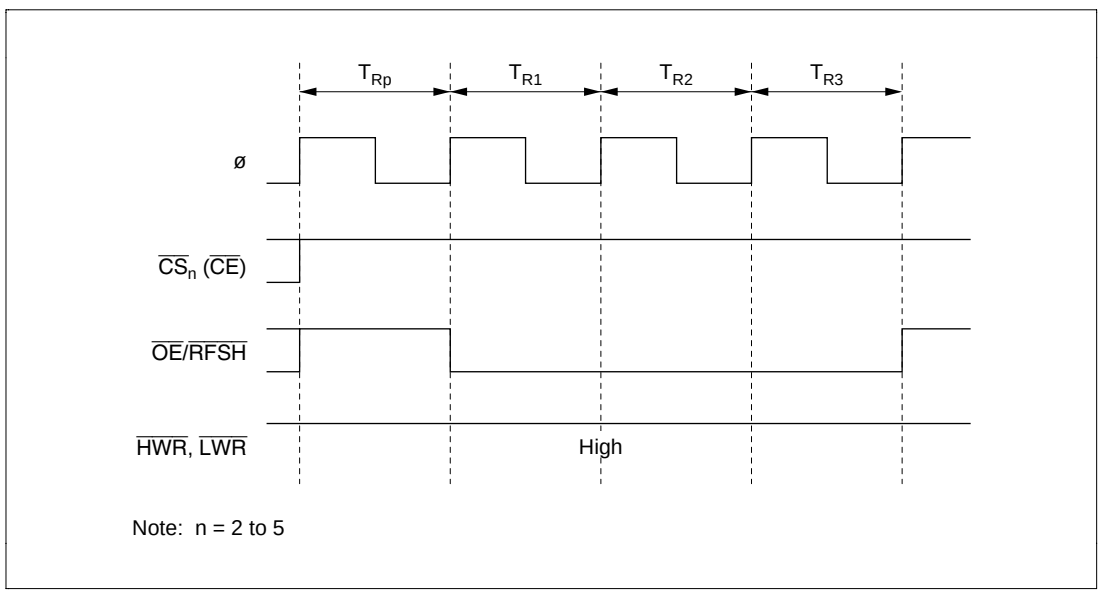



**Figure 6-38 Auto-Refresh Timing**

**(2) Self-Refreshing**

PSRAM is placed in self-refresh mode by holding the RFSH signal low for the prescribed time or longer. To select self-refreshing, set the RFSHE bit and RMODE bit to 1. Self-refresh mode is entered when a SLEEP instruction is executed to enter software standby mode.

When software standby mode is exited, the RMODE bit is cleared to 0 and self-refresh mode is cleared. Check the characteristics of the PSRAM used before making settings after self-refreshing is cleared.

Figure 6-39 shows self-refresh timing.

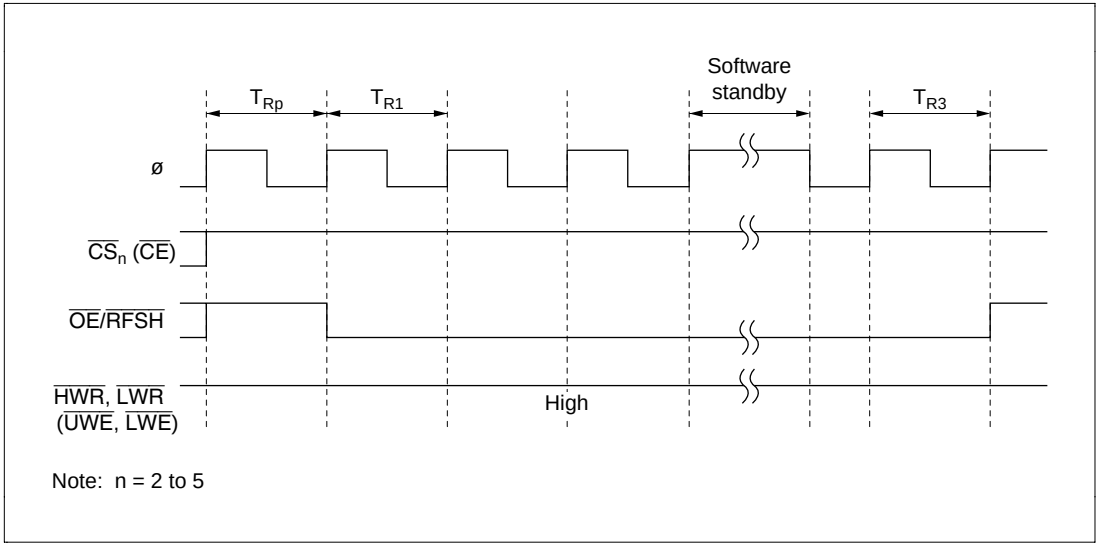



**Figure 6-39 Self-Refresh Timing**

### 6.6.10 Power-On Sequence

A power-on reset initializes the bus controller. If PSRAM is connected, you should check its characteristics and perform the necessary processing.

## 6.7 DMAC Single Address Mode and DRAM/PSRAM Interface

When burst mode is selected with the DRAM or PSRAM interface, the $\overline{\text{DACK}}$ output timing can be selected with the DDS bit. When DRAM or PSRAM space is accessed in DMAC single address mode at the same time, whether or not burst access is to be performed is selected.

### 6.7.1 When DDS = 1

Burst access is performed by determining the address only, irrespective of the bus master. The $\overline{\text{DACK}}$ output goes low from the $T_{C1}$ state in the case of the DRAM interface, and from the $T_2$ state in the case of the PSRAM interface.

Figure 6-40 shows the $\overline{\text{DACK}}$ output timing for the DRAM interface when DDS = 1.

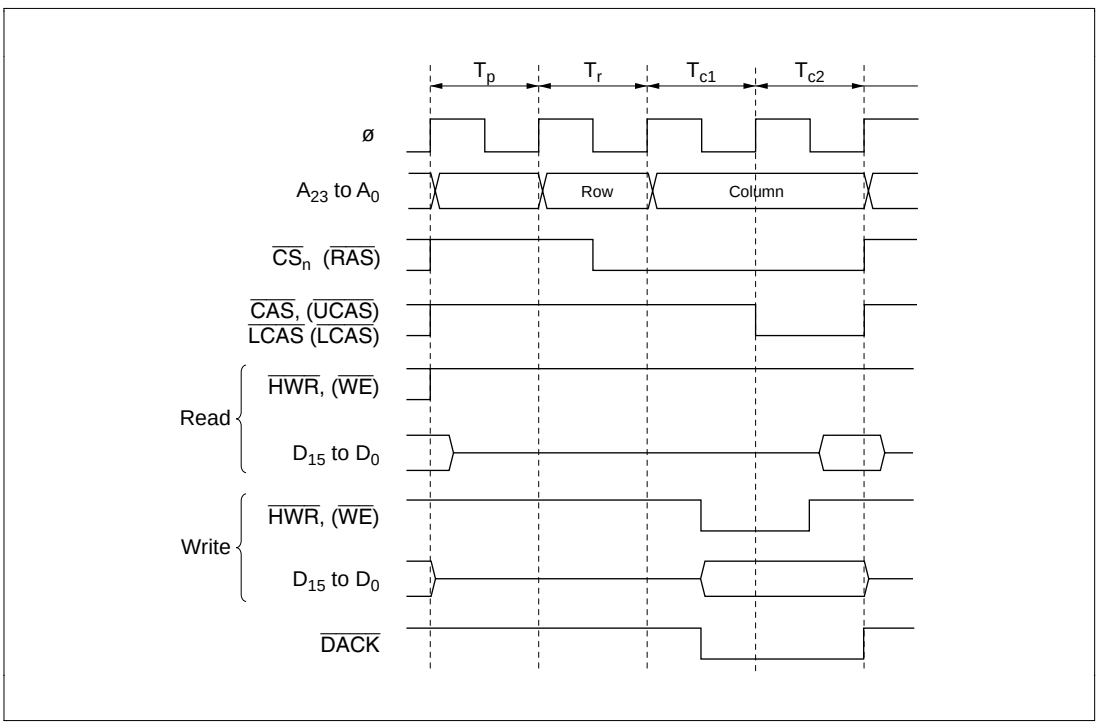



**Figure 6-40 $\overline{\text{DACK}}$ Output Timing when DDS = 1 (Example of DRAM Access)**

### 6.7.2 When DDS = 0

When DRAM or PSRAM space is accessed in DMAC single address mode, full access (normal access) is always performed. The $\overline{\text{DACK}}$ output goes low from the $T_r$ state in the case of the DRAM interface, and from the $T_1$ state in the case of the PSRAM interface.

In modes other than DMAC single address mode, burst access can be used when accessing DRAM or PSRAM space.

Figure 6-41 shows the $\overline{\text{DACK}}$ output timing for the DRAM interface when DDS = 0.

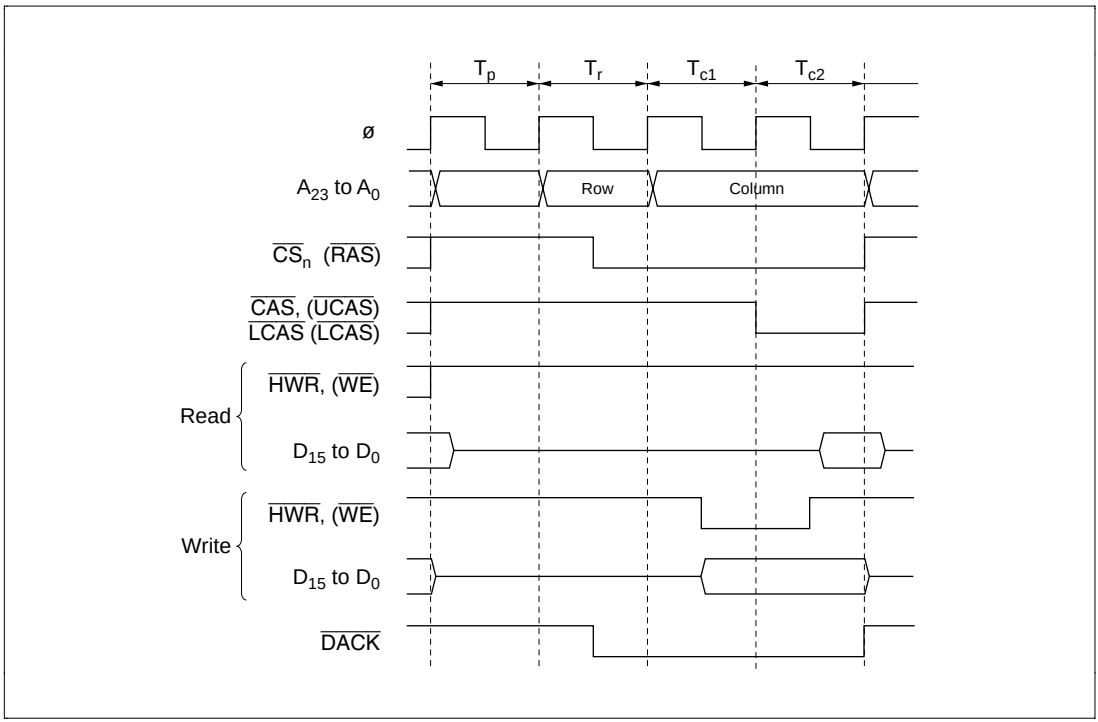



**Figure 6-41 $\overline{\text{DACK}}$ Output Timing when DDS = 0 (Example of DRAM Access)**

## 6.8 Burst ROM Interface

### 6.8.1 Overview

With the H8S/2655 Series, external space area 0 can be designated as burst ROM space, and burst ROM interfacing can be performed. The burst ROM space interface enables 16-bit configuration ROM with burst access capability to be accessed at high speed.

Area 0 can be designated as burst ROM space by means of the BRSTRM bit in BCRH. Consecutive burst accesses of a maximum of 4 words or 8 words can be performed for CPU instruction fetches only. One or two states can be selected for burst access.

Do not select the burst ROM interface and pseudo-SRAM burst operation at the same time.

### 6.8.2 Basic Timing

The number of states in the initial cycle (full access) of the burst ROM interface is in accordance with the setting of the AST0 bit in ASTCR. Also, when the AST0 bit is set to 1, wait state insertion is possible. One or two states can be selected for the burst cycle, according to the setting of the BRSTS1 bit in BCRH. Wait states cannot be inserted. When area 0 is designated as burst ROM space, it becomes 16-bit access space regardless of the setting of the ABW0 bit in ABWCR.

When the BRSTS0 bit in BCRH is cleared to 0, burst access of up to 4 words is performed; when the BRSTS0 bit is set to 1, burst access of up to 8 words is performed.

The basic access timing for burst ROM space is shown in figure 6-42 (a) and (b). The timing shown in figure 6-42 (a) is for the case where the AST0 and BRSTS1 bits are both set to 1, and that in figure 6-42 (b) is for the case where both these bits are cleared to 0.

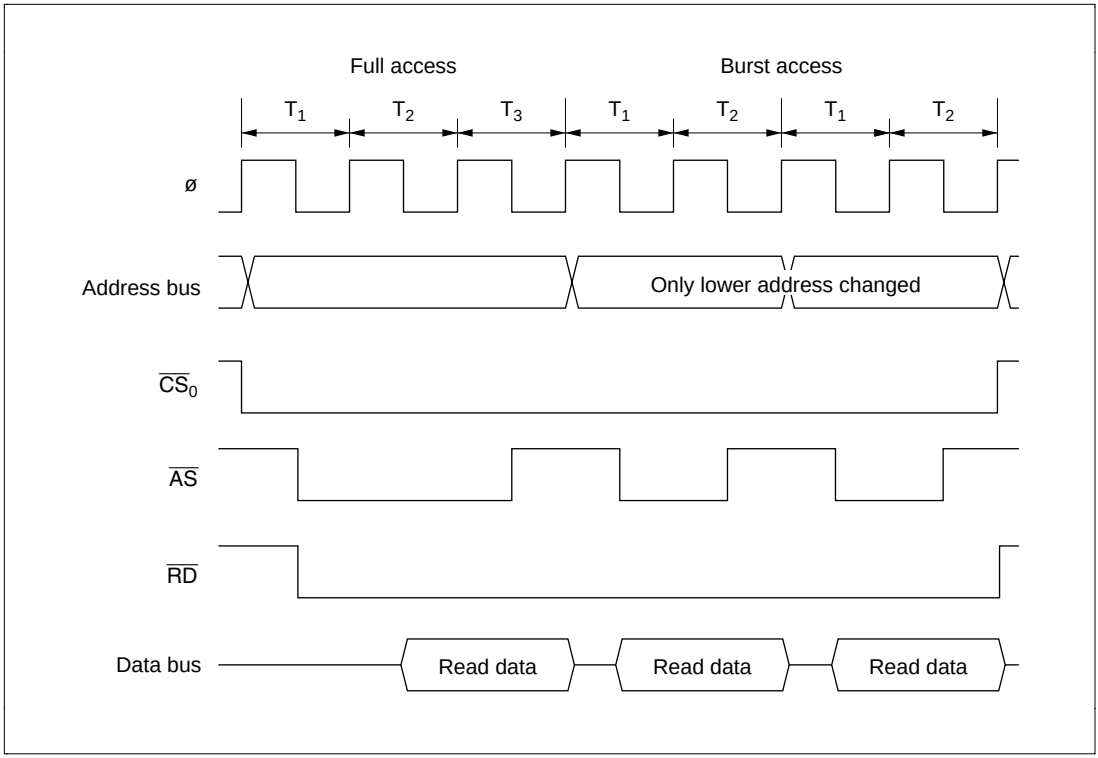



**Figure 6-42 (a) Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 1)**

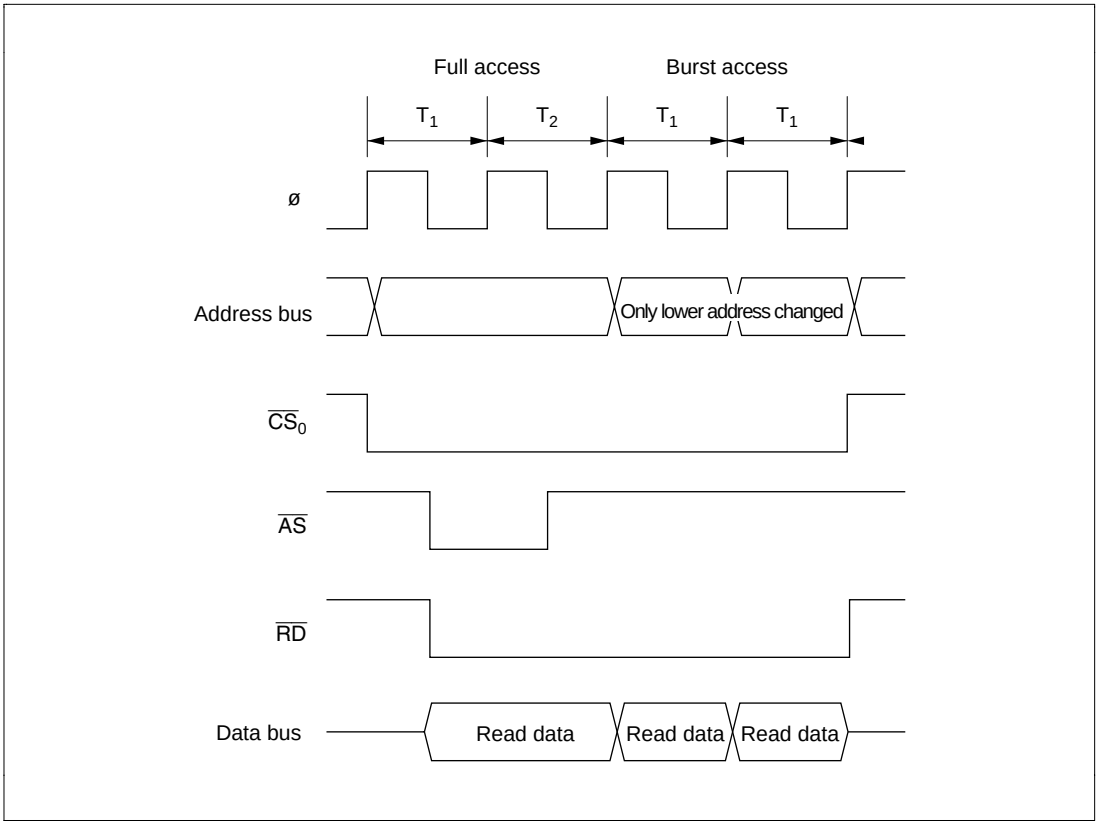



**Figure 6-42 (b) Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 0)**

### 6.8.3 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion using the $\overline{\text{WAIT}}$ pin can be used in the initial cycle (full access) of the burst ROM interface. See section 6.4.5, Wait Control.

Wait states cannot be inserted in a burst cycle.

## 6.9 Idle Cycle

### 6.9.1 Operation

When the H8S/2655 Series accesses external space , it can insert a 1-state idle cycle ($T_I$) between bus cycles in the following two cases: (1) when read accesses between different areas occur consecutively, and (2) when a write cycle occurs immediately after a read cycle. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM, with a long output floating time, and high-speed memory, I/O interfaces, and so on.

**(1) Consecutive Reads between Different Areas**

If consecutive reads between different areas occur while the ICIS1 bit in BCRH is set to 1, an idle cycle is inserted at the start of the second read cycle. This is enabled in advanced mode.

Figure 6-43 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a read cycle from SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

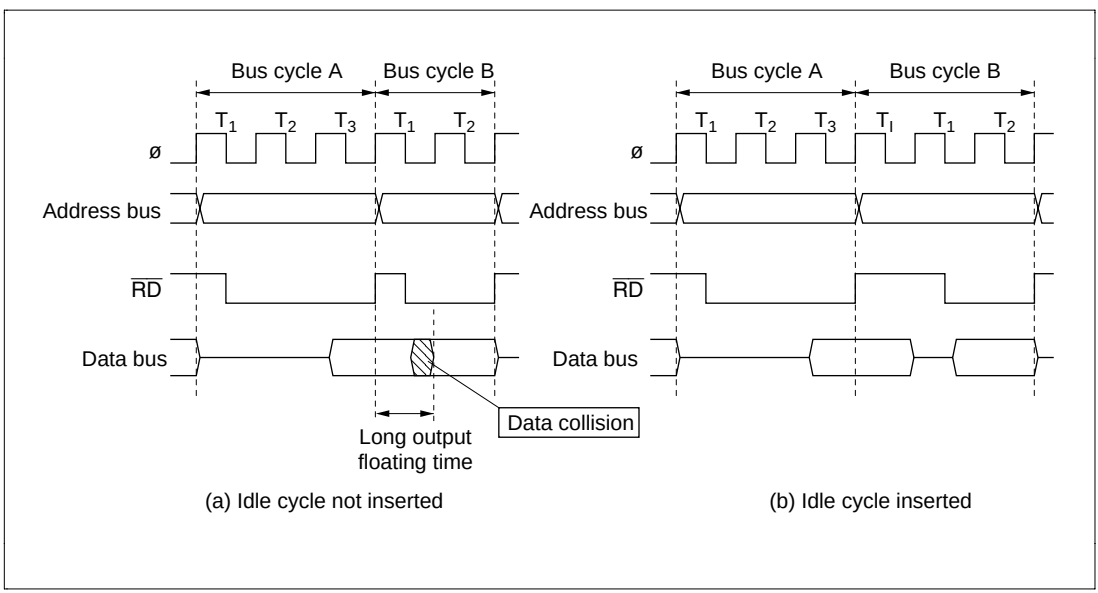



**Figure 6-43 Example of Idle Cycle Operation (1) (When ICIS1 = 1)**

**(2) Write after Read**

If an external write occurs after an external read while the ICIS0 bit in BCRH is set to 1, an idle cycle is inserted at the start of the write cycle. This is enabled in advanced mode and normal mode.

Figure 6-44 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.

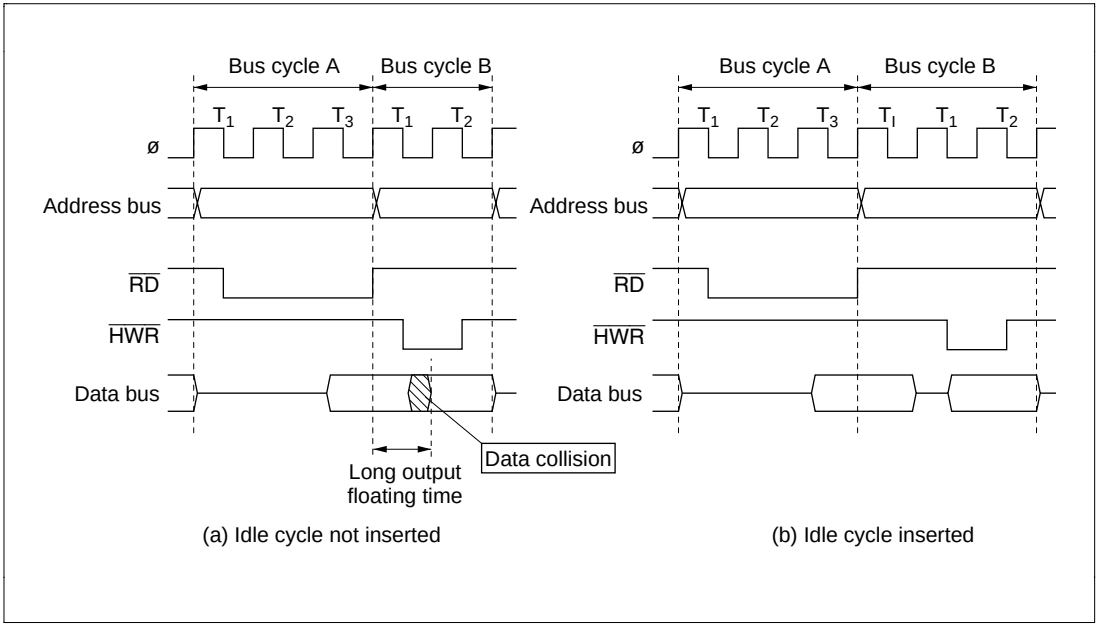



**Figure 6-44 Example of Idle Cycle Operation (2) (When ICIS0 = 1)**

**(3) Usage Notes**

When DRAM space is accessed, the ICIS0 and ICIS1 bit settings are disabled. In the case of consecutive reads between different areas, for example, if the second access is a DRAM access, only a $T_p$ cycle is inserted, and a $T_I$ cycle is not. The timing in this case is shown in figure 6-45.

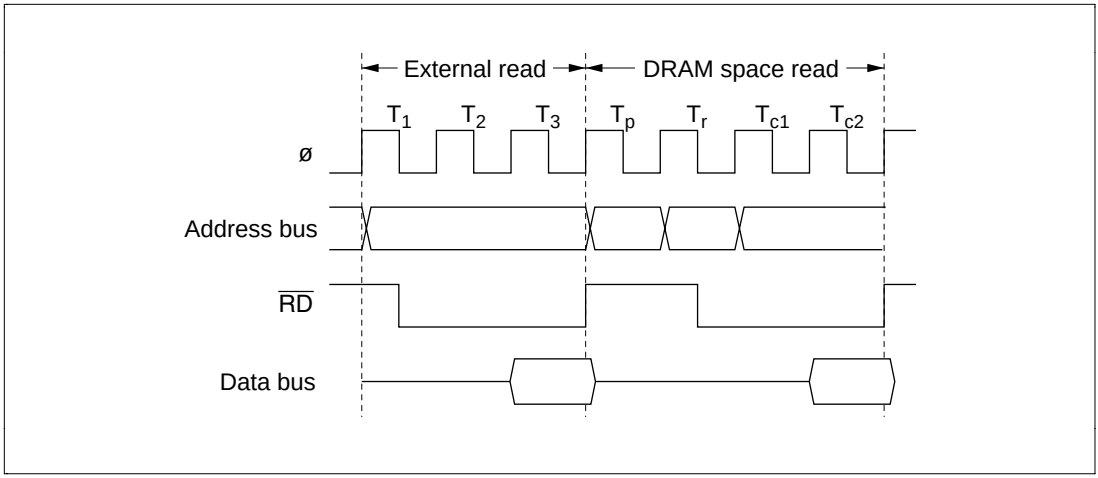



**Figure 6-45 Example of DRAM Access after External Read**

### 6.9.2 Pin States in Idle Cycle

Table 6-10 shows pin states in an idle cycle.

**Table 6-10 Pin States in Idle Cycle**

| Pins | Pin State |
|---|---|
| $A_{23}$ to $A_0$ | Contents of next bus cycle |
| $D_{15}$ to $D_0$ | High impedance |
| $\overline{CS}_n$ | High*1 |
| $\overline{CAS}/\overline{OE}$ | High*2 |
| $\overline{AS}$ | High |
| $\overline{RD}$ | High |
| $\overline{HWR}$ | High |
| $\overline{LWR}$ | High |
| $\overline{DACK}_n$ | High |

Notes: 1. Remains low in PSRAM space CS down mode. Also remains low in DRAM space RAS down mode or a refresh cycle.
2. Remains low in PSRAM space CS down mode or a refresh cycle.

## 6.10 Write Data Buffer Function

The H8S/2655 Series has a write data buffer function in the external data bus. Using the write data buffer function enables external writes and DMA single address mode transfers to be executed in parallel with internal accesses. The write data buffer function is made available by setting the WDBE bit in BCRL to 1.

Figure 6-46 shows an example of the timing when the write data buffer function is used. When this function is used, if an external write or DMA single address mode transfer continues for 2 states or longer, and there is an internal access next, only an external write is executed in the first state, but from the next state onward an internal access (on-chip memory or internal I/O register read/write) is executed in parallel with the external write rather than waiting until it ends.

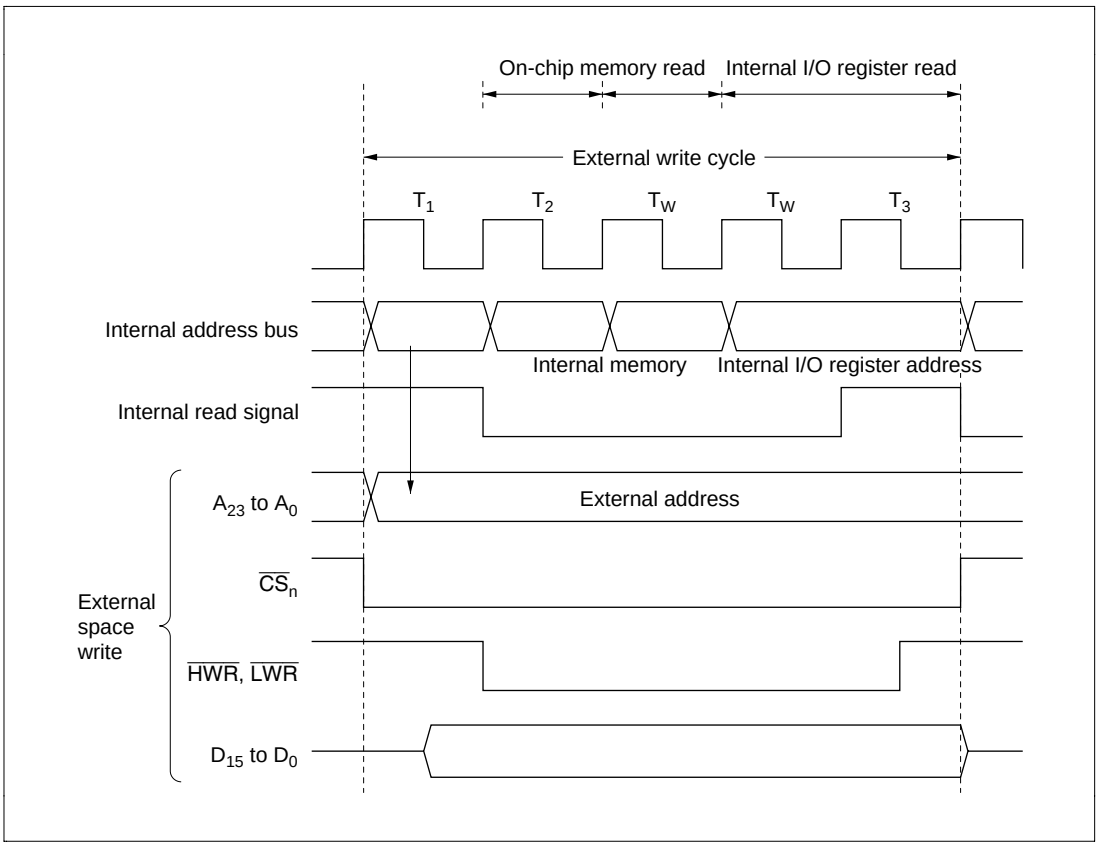



**Figure 6-46 Example of Timing when Write Data Buffer Function is Used**

## 6.11 Bus Release

### 6.11.1 Overview

The H8S/2655 Series can release the external bus in response to a bus request from an external device. In the external bus released state, the internal bus master continues to operate as long as there is no external access.

If an internal bus master wants to make an external access in the external bus released state, or if a refresh request is generated, it can issue a bus request off-chip.

### 6.11.2 Operation

In external expansion mode, the bus can be released to an external device by setting the BRLE bit in BCRL to 1. Driving the $\overline{\text{BREQ}}$ pin low issues an external bus request to the H8S/2655 Series. When the $\overline{\text{BREQ}}$ pin is sampled, at the prescribed timing the $\overline{\text{BACK}}$ pin is driven low, and the address bus, data bus, and bus control signals are placed in the high-impedance state, establishing the external bus-released state.

In the external bus released state, an internal bus master can perform accesses using the internal bus. When an internal bus master wants to make an external access, it temporarily defers activation of the bus cycle, and waits for the bus request from the external bus master to be dropped. Even if a refresh request is generated in the external bus released state, refresh control is deferred until the external bus master drops the bus request.

If the BREQOE bit in BCRL is set to 1, when an internal bus master wants to make an external access in the external bus released state, or when a refresh request is generated, the $\overline{\text{BREQO}}$ pin is driven low and a request can be made off-chip to drop the bus request.

When the $\overline{\text{BREQ}}$ pin is driven high, the $\overline{\text{BACK}}$ pin is driven high at the prescribed timing and the external bus released state is terminated.

In the event of simultaneous external bus release request, refresh request, and external access request generation, the order of priority is as follows:

(High) Refresh > External bus release > Internal bus master external access (Low)

### 6.11.3 Pin States in External Bus Released State

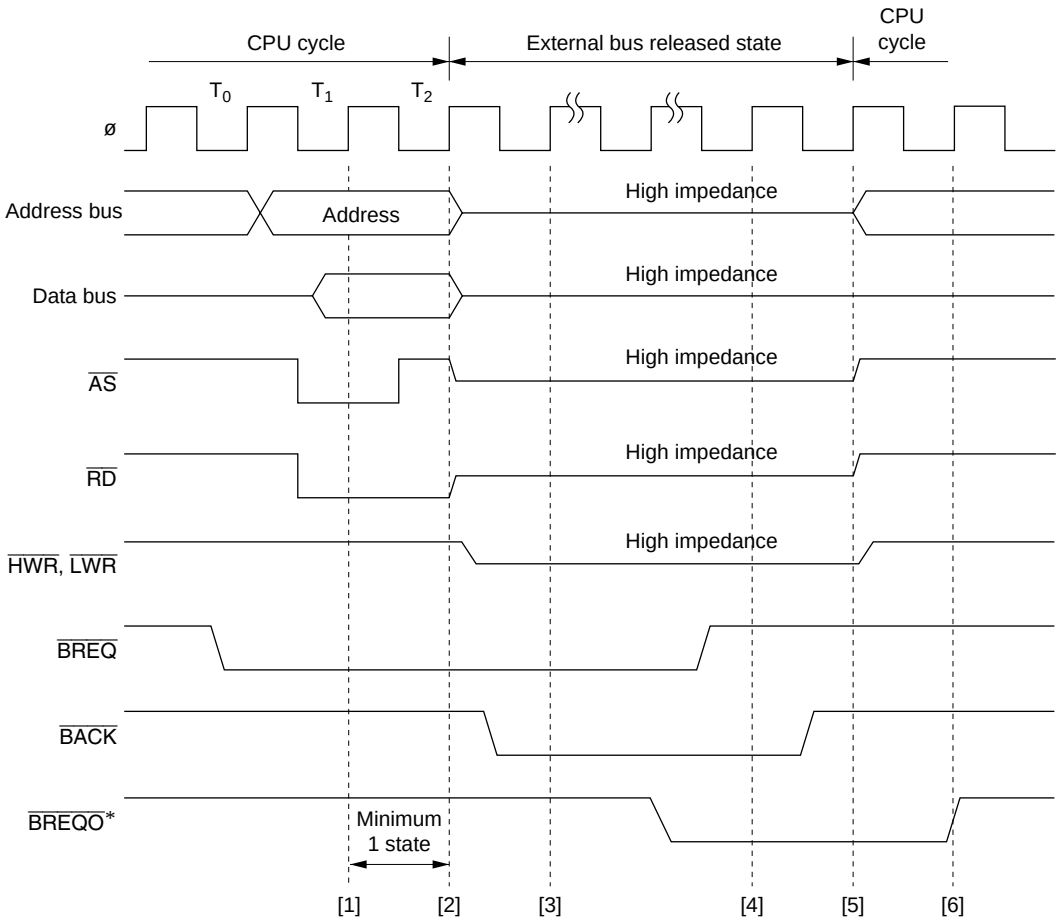
Table 6-11 shows pin states in the external bus released state.

**Table 6-11 Pin States in Bus Released State**

| Pins | Pin State |
|---|---|
| $A_{23}$ to $A_0$ | High impedance |
| $D_{15}$ to $D_0$ | High impedance |
| $\overline{CS}_n$ | High impedance |
| $\overline{CAS}$ | High impedance |
| $\overline{AS}$ | High impedance |
| $\overline{RD}$ | High impedance |
| $\overline{HWR}$ | High impedance |
| $\overline{LWR}$ | High impedance |
| $\overline{DACK}_n$ | High |

### 6.11.4 Transition Timing

Figure 6-47 shows the timing for transition to the bus-released state.




[1] Low level of $\overline{\text{BREQ}}$ pin is sampled at rise of $T_2$ state.

[2] $\overline{\text{BACK}}$ pin is driven low at end of CPU read cycle, releasing bus to external bus master.

[3] $\overline{\text{BREQ}}$ pin state is still sampled in external bus released state.

[4] High level of $\overline{\text{BREQ}}$ pin is sampled.

[5] $\overline{\text{BACK}}$ pin is driven high, ending bus release cycle.

[6] $\overline{\text{BREQO}}$ signal goes high 1.5 clocks after $\overline{\text{BACK}}$ signal goes high.

Note: * Output only when BREQOE is set to 1.

**Figure 6-47 Bus-Released State Transition Timing**

## 6.12 Bus Arbitration

### 6.12.1 Overview

The H8S/2655 Series has a bus arbiter that arbitrates bus master operations.

There are three bus masters, the CPU, DTC, and DMAC, which perform read/write operations when they have possession of the bus. Each bus master requests the bus by means of a bus request signal. The bus arbiter determines priorities at the prescribed timing, and permits use of the bus by means of a bus request acknowledge signal. The selected bus master then takes possession of the bus and begins its operation.

### 6.12.2 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master making the request. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The order of priority of the bus masters is as follows:

(High) DMAC > DTC > CPU (Low)

An internal bus access by an internal bus master, external bus release, and refreshing, can be executed in parallel.

In the event of simultaneous external bus release request, refresh request, and internal bus master external access request generation, the order of priority is as follows:

(High) Refresh > External bus release > Internal bus master external access (Low)

### 6.12.3 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific times at which each bus master can relinquish the bus.

**CPU:** The CPU is the lowest-priority bus master, and if a bus request is received from the DTC or DMAC, the bus arbiter transfers the bus to the bus master that issued the request. The timing for transfer of the bus is as follows:

- The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the operations. See Appendix A-5, Bus States During Instruction Execution, for timings at which the bus is not transferred.
- If the CPU is in sleep mode, it transfers the bus immediately.

**DTC:** The DTC sends the bus arbiter a request for the bus when an activation request is generated.

The DTC can release the bus after a vector read, a register information read (3 states), a single data transfer, or a register information write (3 states). It does not release the bus during a register information read (3 states), a single data transfer, or a register information write (3 states).

**DMAC:** The DMAC sends the bus arbiter a request for the bus when an activation request is generated.

In the case of an external request in short address mode or normal mode, and in cycle steal mode, the DMAC releases the bus after a single transfer.

In block transfer mode, it releases the bus after transfer of one block, and in burst mode, after completion of a transfer.

### 6.12.4 External Bus Release Usage Note

External bus release can be performed on completion of an external bus cycle. The $\overline{RD}$ signal, DRAM interface $\overline{RAS}$ and $\overline{CAS}$ signals, and PSRAM interface $\overline{CE}$ and $\overline{OE}$ signals remain low until the end of the external bus cycle. Therefore, when external bus release is performed, the $\overline{RD}$, $\overline{RAS}$, $\overline{CAS}$, $\overline{CE}$, and $\overline{OE}$ signals may change from the low level to the high-impedance state.

## 6.13 Resets and the Bus Controller

In a power-on reset, the H8S/2655, including the bus controller, enters the reset state at that point, and an executing bus cycle is discontinued.

In a manual reset, the bus controller's registers and internal state are maintained, and an executing external bus cycle is completed. In this case, $\overline{\text{WAIT}}$ input is ignored and write data is not guaranteed. Also, since the DMAC is initialized by a manual reset, $\overline{\text{DACK}}$ and $\overline{\text{TEND}}$ output is disabled and these pins become I/O ports controlled by DDR and DR.

# Section 7 DMA Controller

## 7.1 Overview

The H8S/2655 Series has a built-in DMA controller (DMAC) which can carry out data transfer on up to 4 channels.

### 7.1.1 Features

The features of the DMAC are listed below.

- Choice of short address mode or full address mode

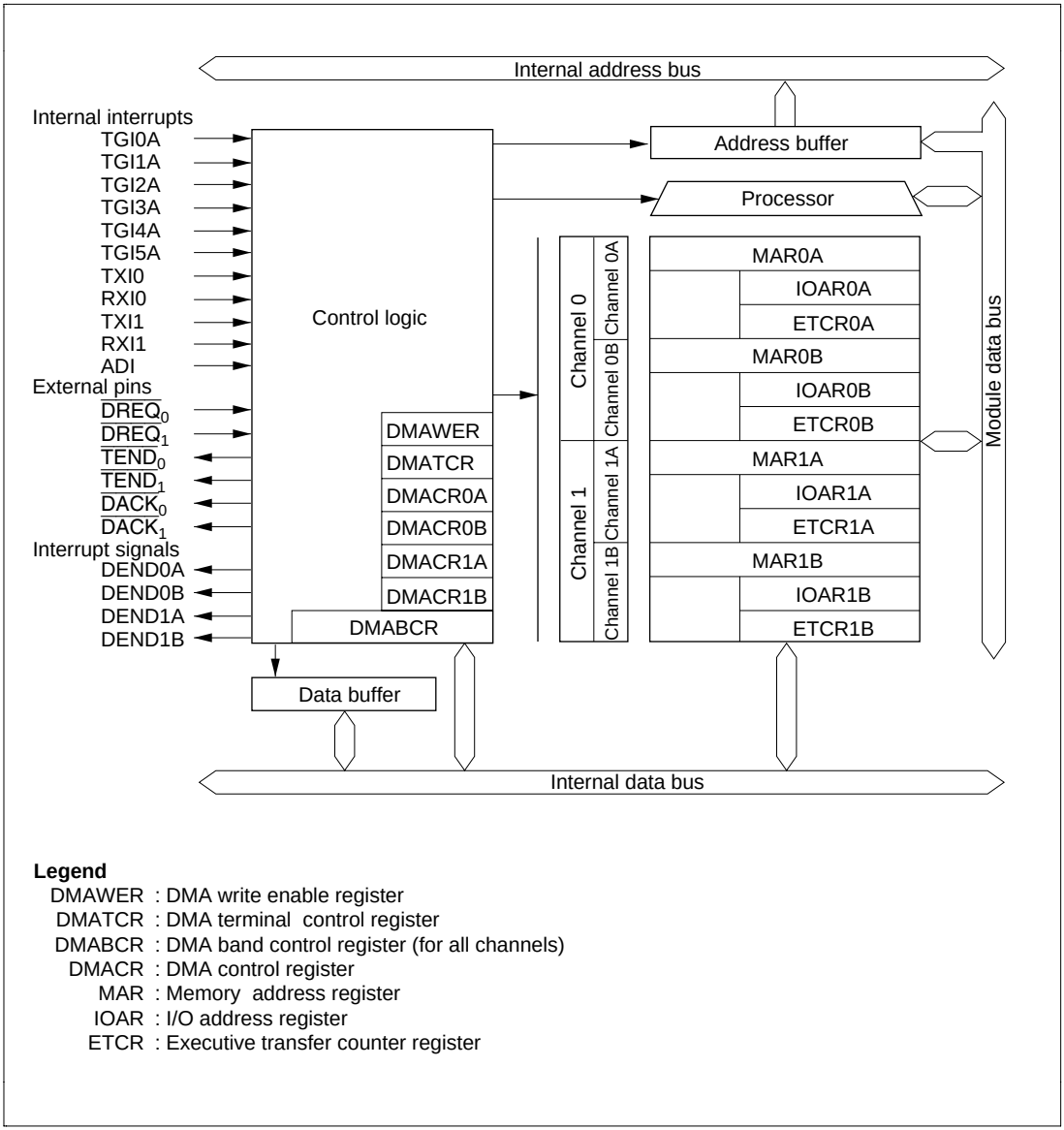
  Short address mode
  - Maximum of 4 channels can be used
  - Choice of dual address mode or single address mode
  - In dual address mode, one of the two addresses, transfer source and transfer destination, is specified as 24 bits and the other as16 bits
  - In single address mode, transfer source or transfer destination address only is specified as 24 bits
  - In single address mode, transfer can be performed in one bus cycle
  - Choice of sequential mode, idle mode, or repeat mode for dual address mode and single address mode

  Full address mode
  - Maximum of 2 channels can be used
  - Transfer source and transfer destination address specified as 24 bits
  - Choice of normal mode or block transfer mode

- 16-Mbyte address space can be specified directly
- Byte or word can be set as the transfer unit
- Activation sources: internal interrupt, external request, auto-request (depending on transfer mode)
  - Six 16-bit timer-pulse unit (TPU) compare match/input capture interrupts
  - Serial communication interface (SCI0, SCI1) transmission complete interrupt, reception complete interrupt
  - A/D converter conversion end interrupt
  - External request
  - Auto-request

### 7.1.2 Block Diagram

A block diagram of the DMAC is shown in figure 7-1.




**Legend**

DMAWER : DMA write enable register
DMATCR : DMA terminal control register
DMABCR : DMA band control register (for all channels)
DMACR : DMA control register
MAR : Memory address register
IOAR : I/O address register
ETCR : Executive transfer counter register

**Figure 7-1 Block Diagram of DMAC**

### 7.1.3 Overview of Functions

Tables 7-1 (1) and (2) summarize DMAC functions in short address mode and full address mode, respectively.

**Table 7-1 (1) Overview of DMAC Functions (Short Address Mode)**

| | | Address Register Bit Length | |
|---|---|---|---|
| **Transfer Mode** | **Transfer Source** | **Source** | **Destination** |
| Dual address mode<br>• Sequential mode<br>— 1-byte or 1-word transfer executed for one transfer request<br>— Memory address incremented/decremented by 1 or 2<br>— 1 to 65536 transfers<br>• Idle mode<br>— 1-byte or 1-word transfer executed for one transfer request<br>— Memory address fixed<br>— 1 to 65536 transfers<br>• Repeat mode<br>— 1-byte or 1-word transfer executed for one transfer request<br>— Memory address incremented/ decremented by 1 or 2<br>— After specified number of transfers (1 to 256), initial state is restored and operation continues | • TPU channel 0 to 5 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• A/D converter conversion end interrupt<br>• External request | 24/16 | 16/24 |
| Single address mode<br>• 1-byte or 1-word transfer executed for one transfer request<br>• Transfer in 1 bus cycle using $\overline{\text{DACK}}$ pin in place of address specifying I/O<br>• Specifiable for modes (1) to (3) | • External request | 24<br>/$\overline{\text{DACK}}$ | $\overline{\text{DACK}}$<br>/24 |

**Table 7-1 (2) Overview of DMAC Functions (Full Address Mode)**

| | | Address Register Bit Length | |
|---|---|---|---|
| **Transfer Mode** | **Transfer Source** | **Source** | **Destination** |
| • Normal mode<br>Auto-request<br>— Transfer request retained internally<br>— Transfers continue for the specified number of times (1 to 65536)<br>— Choice of burst or cycle steal transfer | • Auto-request | 24 | 24 |
| External request<br>— 1-byte or 1-word transfer executed for one transfer request<br>— 1 to 65536 transfers | • External request | | |
| • Block transfer mode<br>— Specified block size transfer executed for one transfer request<br>— 1 to 65536 transfers<br>— Either source or destination specifiable as block area<br>— Block size: 1 to 256 bytes or words | • TPU channel 0 to 5 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• External request<br>• A/D converter conversion end interrupt | 24 | 24 |

### 7.1.4 Pin Configuration

Table 7-2 summarizes the DMAC pins.

In short address mode, external request transfer, single address transfer, and transfer end output are not performed for channel A.

The DMA transfer acknowledge function is used in channel B single address mode in short address mode.

When the $\overline{\text{DREQ}}$ pin is used, do not designate the corresponding port for output.

With regard to the $\overline{\text{DACK}}$ pins, setting single address transfer automatically sets the corresponding port to output, functioning as a $\overline{\text{DACK}}$ pin.

With regard to the $\overline{\text{TEND}}$ pins, whether or not the corresponding port is used as a $\overline{\text{TEND}}$ pin can be specified by means of a register setting.

**Table 7-2 DMAC Pins**

| Channel | Pin Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | DMA request 0 | $\overline{\text{DREQ}}_0$ | Input | DMAC channel 0 external request |
| | DMA transfer acknowledge 0 | $\overline{\text{DACK}}_0$ | Output | DMAC channel 0 single address transfer acknowledge |
| | DMA transfer end 0 | $\overline{\text{TEND}}_0$ | Output | DMAC channel 0 transfer end |
| 1 | DMA request 1 | $\overline{\text{DREQ}}_1$ | Input | DMAC channel 1 external request |
| | DMA transfer acknowledge 1 | $\overline{\text{DACK}}_1$ | Output | DMAC channel 1 single address transfer acknowledge |
| | DMA transfer end 1 | $\overline{\text{TEND}}_1$ | Output | DMAC channel 1 transfer end |

### 7.1.5 Register Configuration

Table 7-3 summarizes the DMAC registers.

**Table 7-3 DMAC Registers**

| Channel | Address* | Bus Width | Name | Abbreviation | R/W | Initial Value |
|---|---|---|---|---|---|---|
| 0 | H'FEE0 | 16 bits | Memory address register 0A | MAR0A | R/W | Undefined |
| | H'FEE4 | 16 bits | I/O address register 0A | IOAR0A | R/W | Undefined |
| | H'FEE6 | 16 bits | Transfer count register 0A | ETCR0A | R/W | Undefined |
| | H'FEE8 | 16 bits | Memory address register 0B | MAR0B | R/W | Undefined |
| | H'FEEC | 16 bits | I/O address register 0B | IOAR0B | R/W | Undefined |
| | H'FEEE | 16 bits | Transfer count register 0B | ETCR0B | R/W | Undefined |
| 1 | H'FEF0 | 16 bits | Memory address register 1A | MAR1A | R/W | Undefined |
| | H'FEF4 | 16 bits | I/O address register 1A | IOAR1A | R/W | Undefined |
| | H'FEF6 | 16 bits | Transfer count register 1A | ETCR1A | R/W | Undefined |
| | H'FEF8 | 16 bits | Memory address register 1B | MAR1B | R/W | Undefined |
| | H'FEFC | 16 bits | I/O address register 1B | IOAR1B | R/W | Undefined |
| | H'FEFE | 16 bits | Transfer count register 1B | ETCR1B | R/W | Undefined |
| 0, 1 | H'FF00 | 8 bits | DMA write enable register | DMAWER | R/W | H'00 |
| | H'FF01 | 8 bits | DMA terminal control register | DMATCR | R/W | H'00 |
| | H'FF02 | 16 bits | DMA control register 0A | DMACR0A | R/W | H'00 |
| | H'FF03 | 16 bits | DMA control register 0B | DMACR0B | R/W | H'00 |
| | H'FF04 | 16 bits | DMA control register 1A | DMACR1A | R/W | H'00 |
| | H'FF05 | 16 bits | DMA control register 1B | DMACR1B | R/W | H'00 |
| | H'FF06 | 16 bits | DMA band control register | DMABCR | R/W | H'0000 |
| | H'FF3C | 8 bits | Module stop control register | MSTPCR | R/W | H'3FFF |

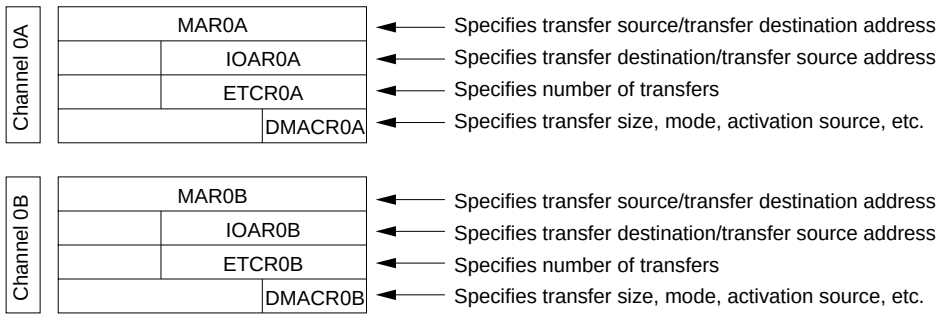
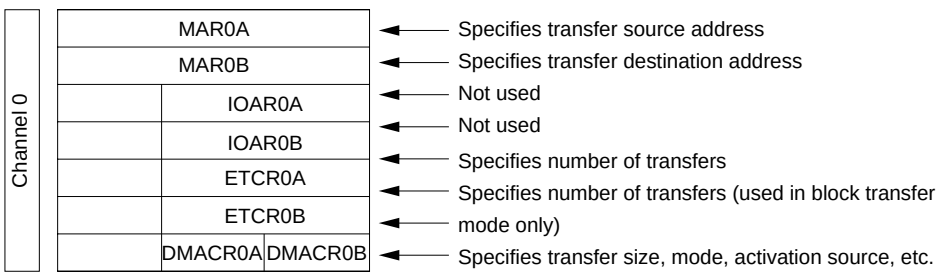
Note: * Lower 16 bits of the address.

## 7.2 Register Descriptions (1) (Short Address Mode)

Short address mode transfer can be performed for channels A and B independently.

Short address mode transfer is specified for each channel by clearing the FAE bit in DMABCR to 0, as shown in table 7-4. Short address mode or full address mode can be selected for channels 1 and 0 independently by means of bits FAE1 and FAE0.

**Table 7-4 Short Address Mode and Full Address Mode (For 1 Channel: Example of Channel 0)**

| FAE0 | Description |
|---|---|
| 0 | Short address mode specified (channels A and B operate independently)<br>  |
| 1 | Full address mode specified (channels A and B operate in combination)<br>  |

### 7.2.1 Memory Address Registers (MAR)

| Bit | : | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR | : | — | — | — | — | — | — | — | — | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| R/W | : | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

MAR is a 32-bit readable/writable register that specifies the transfer source address or destination address.

The upper 8 bits of MAR are reserved: they are always read as 0, and cannot be modified.

Whether MAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

MAR is incremented or decremented each time a byte or word transfer is executed, so that the address specified by MAR is constantly updated. For details, see section 7.2.4, DMA Control Register (DMACR).

MAR is not initialized by a reset or in standby mode.

### 7.2.2 I/O Address Register (IOAR)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IOAR | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

IOAR is a 16-bit readable/writable register that specifies the lower 16 bits of the transfer source address or destination address. The upper 8 bits of the transfer address are automatically set to H'FF.

Whether IOAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

IOAR is invalid in single address mode.

IOAR is not incremented or decremented each time a transfer is executed, so that the address specified by IOAR is fixed.

IOAR is not initialized by a reset or in standby mode.

### 7.2.3 Execute Transfer Count Register (ETCR)

ETCR is a 16-bit readable/writable register that specifies the number of transfers. The setting of this register is different for sequential mode and idle mode on the one hand, and for repeat mode on the other.

#### (1) Sequential Mode and Idle Mode

**Transfer Counter**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

In sequential mode and idle mode, ETCR functions as a 16-bit transfer counter (with a count range of 1 to 65536). ETCR is decremented by 1 each time a transfer is performed, and when the count reaches H'0000, the DTE bit in DMABCR is cleared, and transfer ends.

**(2) Repeat Mode**

**Transfer Number Storage**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| ETCRH | : | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Transfer Counter**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ETCRL | : | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

In repeat mode, ETCR functions as transfer counter ETCRL (with a count range of 1 to 256) and transfer number storage register ETCRH. ETCRL is decremented by 1 each time a transfer is performed, and when the count reaches H'00, ETCRL is loaded with the value in ETCRH. At this point, MAR is automatically restored to the value it had when the count was started. The DTE bit in DMABCR is not cleared, and so transfers can be performed repeatedly until the DTE bit is cleared by the user.

ETCR is not initialized by a reset or in standby mode.

### 7.2.4 DMA Control Register (DMACR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMACR | : | DTSZ | DTID5 | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMACR is an 8-bit readable/writable register that controls the operation of each DMAC channel.

DMACR is initialized to H'00 by a reset, and in standby mode.

**Bit 7—Data Transfer Size (DTSZ):** Selects the size of data to be transferred at one time.

| Bit 7 | | |
|---|---|---|
| **DTSZ** | **Description** | |
| 0 | Byte-size transfer | (Initial value) |
| 1 | Word-size transfer | |

**Bit 6—Data Transfer Increment/Decrement (DTID):** Selects incrementing or decrementing of MAR every data transfer in sequential mode or repeat mode.

In idle mode, MAR is neither incremented nor decremented.

| Bit 6 | | |
|---|---|---|
| **DTID** | **Description** | |
| 0 | MAR is incremented after a data transfer | (Initial value) |
| | • When DTSZ = 0, MAR is incremented by 1 after a transfer | |
| | • When DTSZ = 1, MAR is incremented by 2 after a transfer | |
| 1 | MAR is decremented after a data transfer | |
| | • When DTSZ = 0, MAR is decremented by 1 after a transfer | |
| | • When DTSZ = 1, MAR is decremented by 2 after a transfer | |

**Bit 5—Repeat Enable (RPE):** Used in combination with the DTIE bit in DMABCR to select the mode (sequential, idle, or repeat) in which transfer is to be performed.

| Bit 5 | DMABCR | | |
|---|---|---|---|
| **RPE** | **DTIE** | **Description** | |
| 0 | 0 | Transfer in sequential mode (no transfer end interrupt) | (Initial value) |
| | 1 | Transfer in sequential mode (with transfer end interrupt) | |
| 1 | 0 | Transfer in repeat mode (no transfer end interrupt) | |
| | 1 | Transfer in idle mode (with transfer end interrupt) | |

For details of operation in sequential, idle, and repeat mode, see section 7.5.2, Sequential Mode, section 7.5.3, Idle Mode, and section 7.5.4, Repeat Mode.

**Bit 4—Data Transfer Direction (DTDIR):** Used in combination with the SAE bit in DMABCR to specify the data transfer direction (source or destination). The function of this bit is therefore different in dual address mode and single address mode.

| DMABCR | Bit 4 | |
|---|---|---|
| **SAE** | **DTDIR** | **Description** |
| 0 | 0 | Transfer with MAR as source address and IOAR as destination address (Initial value) |
| | 1 | Transfer with IOAR as source address and MAR as destination address |
| 1 | 0 | Transfer with MAR as source address and $\overline{\text{DACK}}$ pin as write strobe |
| | 1 | Transfer with $\overline{\text{DACK}}$ pin as read strobe and MAR as destination address |

**Bits 3 to 0—Data Transfer Factor (DTF3 to DTF0):** These bits select the data transfer factor (activation source). There are some differences in activation sources for channel A and for channel B.

**Channel A**

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| **DTF3** | **DTF2** | **DTF1** | **DTF0** | **Description** |
| 0 | 0 | 0 | 0 | — (Initial value) |
| | | | 1 | Activated by A/D converter conversion end interrupt |
| | | 1 | 0 | — |
| | | | 1 | — |
| | 1 | 0 | 0 | Activated by SCI channel 0 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 0 reception complete interrupt |
| | | 1 | 0 | Activated by SCI channel 1 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 1 reception complete interrupt |
| 1 | 0 | 0 | 0 | Activated by TPU channel 0 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 1 compare match/input capture A interrupt |
| | | 1 | 0 | Activated by TPU channel 2 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 3 compare match/input capture A interrupt |
| | 1 | 0 | 0 | Activated by TPU channel 4 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 5 compare match/input capture A interrupt |
| | | 1 | 0 | — |
| | | | 1 | — |

**Channel B**

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| **DTF3** | **DTF2** | **DTF1** | **DTF0** | **Description** |
| 0 | 0 | 0 | 0 | — (Initial value) |
| | | | 1 | Activated by A/D converter conversion end interrupt |
| | | 1 | 0 | Activated by $\overline{\text{DREQ}}$ pin falling edge input* |
| | | | 1 | Activated by $\overline{\text{DREQ}}$ pin low-level input |
| | 1 | 0 | 0 | Activated by SCI channel 0 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 0 reception complete interrupt |
| | | 1 | 0 | Activated by SCI channel 1 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 1 reception complete interrupt |
| 1 | 0 | 0 | 0 | Activated by TPU channel 0 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 1 compare match/input capture A interrupt |
| | | 1 | 0 | Activated by TPU channel 2 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 3 compare match/input capture A interrupt |
| | 1 | 0 | 0 | Activated by TPU channel 4 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 5 compare match/input capture A interrupt |
| | | 1 | 0 | — |
| | | | 1 | — |

Note: * Detected as a low level in the first transfer after transfer is enabled.

The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 7.5.13, DMAC Multi-Channel Operation.

### 7.2.5 DMA Band Control Register (DMABCR)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRH | : | FAE1 | FAE0 | SAE1 | SAE0 | DTA1B | DATA1A | DTA0B | DTA0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRL | : | DTE1B | DTE1A | DTE0B | DTE0A | DTIE1B | DTIE1A | DTIE0B | DTIE0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMABCR is a 16-bit readable/writable register that controls the operation of each DMAC channel.

DMABCR is initialized to H'0000 by a reset, and in standby mode.

**Bit 15—Full Address Enable 1 (FAE1):** Specifies whether channel 1 is to be used in short address mode or full address mode.

In short address mode, channels 1A and 1B are used as independent channels.

**Bit 15**

| FAE1 | Description | |
|---|---|---|
| 0 | Short address mode | (Initial value) |
| 1 | Full address mode | |

**Bit 14—Full Address Enable 0 (FAE0):** Specifies whether channel 0 is to be used in short address mode or full address mode.

In short address mode, channels 0A and 0B are used as independent channels.

**Bit 14**

| FAE0 | Description | |
|---|---|---|
| 0 | Short address mode | (Initial value) |
| 1 | Full address mode | |

**Bit 13—Single Address Enable 1 (SAE1):** Specifies whether channel 1B is to be used for transfer in dual address mode or single address mode.

**Bit 13**

| SAE1 | Description | |
|---|---|---|
| 0 | Transfer in dual address mode | (Initial value) |
| 1 | Transfer in single address mode | |

This bit is invalid in full address mode.

**Bit 12—Single Address Enable 0 (SAE0):** Specifies whether channel 0B is to be used for transfer in dual address mode or single address mode.

**Bit 12**

| SAE0 | Description | |
|---|---|---|
| 0 | Transfer in dual address mode | (Initial value) |
| 1 | Transfer in single address mode | |

This bit is invalid in full address mode.

**Bits 11 to 8—Data Transfer Acknowledge (DTA):** These bits enable or disable clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.

When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU or DTC.

When DTE = 1 and DTA = 0, the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU or DTC in parallel. In this case, the interrupt source should be cleared by the CPU or DTC transfer.

When DTE = 0, the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU or DTC regardless of the DTA bit setting.

**Bit 11—Data Transfer Acknowledge 1B (DTA1B):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1B data transfer factor setting.

**Bit 11**

| DTA1B | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bit 10—Data Transfer Acknowledge 1A (DTA1A):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1A data transfer factor setting.

**Bit 10**

| DTA1A | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bit 9—Data Transfer Acknowledge 0B (DTA0B):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0B data transfer factor setting.

**Bit 9**

| DTA0B | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bit 8—Data Transfer Acknowledge 0A (DTA0A):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0A data transfer factor setting.

**Bit 8**

| DTA0A | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bits 7 to 4—Data Transfer Enable (DTE):** When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU or DTC. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

The conditions for the DTE bit being cleared to 0 are as follows:

- When initialization is performed
- When the specified number of transfers have been completed in a transfer mode other than repeat mode
- When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason

When DTE = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed.

The condition for the DTE bit being set to 1 is as follows:

- When 1 is written to the DTE bit after the DTE bit is read as 0

**Bit 7—Data Transfer Enable 1B (DTE1B):** Enables or disables data transfer on channel 1B.

**Bit 7**

| DTE1B | Description | |
|---|---|---|
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bit 6—Data Transfer Enable 1A (DTE1A):** Enables or disables data transfer on channel 1A.

**Bit 6**

| DTE1A | Description | |
|---|---|---|
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bit 5—Data Transfer Enable 0B (DTE0B):** Enables or disables data transfer on channel 0B.

| Bit 5 | | |
|---|---|---|
| **DTE0B** | **Description** | |
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bit 4—Data Transfer Enable 0A (DTE0A):** Enables or disables data transfer on channel 0A.

| Bit 4 | | |
|---|---|---|
| **DTE0A** | **Description** | |
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bits 3 to 0—Data Transfer End Interrupt Enable (DTIE):** These bits enable or disable an interrupt to the CPU or DTC when transfer ends. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

A transfer end interrupt can be canceled either by clearing the DTIE bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the transfer counter and address register again, and then setting the DTE bit to 1.

**Bit 3—Data Transfer Interrupt Enable 1B (DTIE1B):** Enables or disables the channel 1B transfer end interrupt.

| Bit 3 | | |
|---|---|---|
| **DTIE1B** | **Description** | |
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

**Bit 2—Data Transfer Interrupt Enable 1A (DTIE1A):** Enables or disables the channel 1A transfer end interrupt.

| Bit 2 | | |
|---|---|---|
| **DTIE1A** | **Description** | |
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

**Bit 1—Data Transfer Interrupt Enable 0B (DTIE0B):** Enables or disables the channel 0B transfer end interrupt.

**Bit 1**

| DTIE0B | Description | |
|---|---|---|
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

**Bit 0—Data Transfer Interrupt Enable 0A (DTIE0A):** Enables or disables the channel 0A transfer end interrupt.

**Bit 0**

| DTIE0A | Description | |
|---|---|---|
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

## 7.3 Register Descriptions (2) (Full Address Mode)

Full address mode transfer is performed with channels A and B together. For details of full address mode setting, see table 7-4.

### 7.3.1 Memory Address Register (MAR)

| Bit | : | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR | : | — | — | — | — | — | — | — | — | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| R/W | : | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

MAR is a 32-bit readable/writable register; MARA functions as the transfer source address register, and MARB as the destination address register.

MAR is composed of two 16-bit registers, MARH and MARL. The upper 8 bits of MARH are reserved: they are always read as 0, and cannot be modified.

MAR is incremented or decremented each time a byte or word transfer is executed, so that the source or destination memory address can be updated automatically. For details, see section 7.3.4, DMA Control Register (DMACR).

MAR is not initialized by a reset or in standby mode.

### 7.3.2 I/O Address Register (IOAR)

IOAR is not used in full address transfer.

### 7.3.3 Execute Transfer Count Register (ETCR)

ETCR is a 16-bit readable/writable register that specifies the number of transfers. The function of this register is different in normal mode and in block transfer mode.

ETCR is not initialized by a reset or in standby mode.

**(1) Normal Mode**

ETCRA

**Transfer Counter**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

In normal mode, ETCRA functions as a 16-bit transfer counter. ETCRA is decremented by 1 each time a transfer is performed, and transfer ends when the count reaches H'0000. ETCRB is not used at this time.

ETCRB

ETCRB is not used in normal mode.

**(2) Block Transfer Mode**

ETCRA

**Holds block size**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| ETCRAH | : | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Block size counter**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ETCRAL | : | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

*: Undefined

ETCRB

**Block Transfer Counter**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCRB | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

In block transfer mode, ETCRAL functions as an 8-bit block size counter and ETCRAH holds the block size. ETCRAL is decremented each time a 1-byte or 1-word transfer is performed, and when the count reaches H'00, ETCRAL is loaded with the value in ETCRAH. So by setting the block size in ETCRAH and ETCRAL, it is possible to repeatedly transfer blocks consisting of any desired number of bytes or words.

ETCRB functions in block transfer mode, as a 16-bit block transfer counter. ETCRB is decremented by 1 each time a block is transferred, and transfer ends when the count reaches H'0000.

### 7.3.4 DMA Control Register (DMACR)

DMACR is a 16-bit readable/writable register that controls the operation of each DMAC channel. In full address mode, DMACRA and DMACRB have different functions.

DMACR is initialized to H'0000 by a reset, and in standby mode.

DMACRA

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| DMACRA | : | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMACRB

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMACRB | : | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bit 15—Data Transfer Size (DTSZ):** Selects the size of data to be transferred at one time.

| Bit 15 | | |
|---|---|---|
| **DTSZ** | **Description** | |
| 0 | Byte-size transfer | (Initial value) |
| 1 | Word-size transfer | |

**Bit 14—Source Address Increment/Decrement (SAID)**

**Bit 13—Source Address Increment/Decrement Enable (SAIDE):** These bits specify whether source address register MARA is to be incremented, decremented, or left unchanged, when data transfer is performed.

| Bit 14 | Bit 13 | | |
|---|---|---|---|
| **SAID** | **SAIDE** | **Description** | |
| 0 | 0 | MARA is fixed | (Initial value) |
| | 1 | MARA is incremented after a data transfer<br>• When DTSZ = 0, MARA is incremented by 1 after a transfer<br>• When DTSZ = 1, MARA is incremented by 2 after a transfer | |
| 1 | 0 | MARA is fixed | |
| | 1 | MARA is decremented after a data transfer<br>• When DTSZ = 0, MARA is decremented by 1 after a transfer<br>• When DTSZ = 1, MARA is decremented by 2 after a transfer | |

**Bit 12—Block Direction (BLKDIR)**

**Bit 11—Block Enable (BLKE):** These bits specify whether normal mode or block transfer mode is to be used. If block transfer mode is specified, the BLKDIR bit specifies whether the source side or the destination side is to be the block area.

| Bit 12 | Bit 11 | | |
|---|---|---|---|
| **BLKDIR** | **BLKE** | **Description** | |
| 0 | 0 | Transfer in normal mode | (Initial value) |
| | 1 | Transfer in block transfer mode, destination side is block area | |
| 1 | 0 | Transfer in normal mode | |
| | 1 | Transfer in block transfer mode, source side is block area | |

For operation in normal mode and block transfer mode, see section 7.5, Operation.

**Bits 10 to 7—Reserved:** Can be read or written to.

**Bit 6—Destination Address Increment/Decrement (DAID)**

**Bit 5—Destination Address Increment/Decrement Enable (DAIDE):** These bits specify whether destination address register MARB is to be incremented, decremented, or left unchanged, when data transfer is performed.

| Bit 6 | Bit 5 | |
|---|---|---|
| **DAID** | **DAIDE** | **Description** |
| 0 | 0 | MARB is fixed (Initial value) |
| | 1 | MARB is incremented after a data transfer<br>• When DTSZ = 0, MARB is incremented by 1 after a transfer<br>• When DTSZ = 1, MARB is incremented by 2 after a transfer |
| 1 | 0 | MARB is fixed |
| | 1 | MARB is decremented after a data transfer<br>• When DTSZ = 0, MARB is decremented by 1 after a transfer<br>• When DTSZ = 1, MARB is decremented by 2 after a transfer |

**Bit 4—Reserved:** Can be read or written to.

**Bits 3 to 0—Data Transfer Factor (DTF3 to DTF0):** These bits select the data transfer factor (activation source). The factors that can be specified differ between normal mode and block transfer mode.

- Normal Mode

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|---|
| **DTF3** | **DTF2** | **DTF1** | **DTF0** | **Description** | |
| 0 | 0 | 0 | 0 | — | (Initial value) |
| | | | 1 | — | |
| | | 1 | 0 | Activated by $\overline{\text{DREQ}}$ pin falling edge input | |
| | | | 1 | Activated by $\overline{\text{DREQ}}$ pin low-level input | |
| | 1 | 0 | * | — | |
| | | 1 | 0 | Auto-request (cycle steal) | |
| | | | 1 | Auto-request (burst) | |
| 1 | * | * | * | — | |

*: Don't care

- Block Transfer Mode

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| DTF3 | DTF2 | DTF1 | DTF0 | Description |
| 0 | 0 | 0 | 0 | — (Initial value) |
| | | | 1 | Activated by A/D converter conversion end interrupt |
| | | 1 | 0 | Activated by $\overline{\text{DREQ}}$ pin falling edge input* |
| | | | 1 | Activated by $\overline{\text{DREQ}}$ pin low-level input |
| | 1 | 0 | 0 | Activated by SCI channel 0 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 0 reception complete interrupt |
| | | 1 | 0 | Activated by SCI channel 1 transmission complete interrupt |
| | | | 1 | Activated by SCI channel 1 reception complete interrupt |
| 1 | 0 | 0 | 0 | Activated by TPU channel 0 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 1 compare match/input capture A interrupt |
| | | 1 | 0 | Activated by TPU channel 2 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 3 compare match/input capture A interrupt |
| | 1 | 0 | 0 | Activated by TPU channel 4 compare match/input capture A interrupt |
| | | | 1 | Activated by TPU channel 5 compare match/input capture A interrupt |
| | | 1 | 0 | — |
| | | | 1 | — |

Note: * Detected as a low level in the first transfer after transfer is enabled.

The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 7.5.13, DMAC Multi-Channel Operation.

### 7.3.5 DMA Band Control Register (DMABCR)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRH | : | FAE1 | FAE0 | — | — | DTA1 | — | DTA0 | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRL | : | DTME1 | DTE1 | DTME0 | DTE0 | DTIE1B | DTIE1A | DTIE0B | DTIE0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMABCR is a 16-bit readable/writable register that controls the operation of each DMAC channel.

DMABCR is initialized to H'0000 by a reset, and in standby mode.

**Bit 15—Full Address Enable 1 (FAE1):** Specifies whether channel 1 is to be used in short address mode or full address mode.

In full address mode, channels 1A and 1B are used together as a single channel.

**Bit 15**

| FAE1 | Description | |
|---|---|---|
| 0 | Short address mode | (Initial value) |
| 1 | Full address mode | |

**Bit 14—Full Address Enable 0 (FAE0):** Specifies whether channel 0 is to be used in short address mode or full address mode.

In full address mode, channels 0A and 0B are used together as a single channel.

**Bit 14**

| FAE0 | Description | |
|---|---|---|
| 0 | Short address mode | (Initial value) |
| 1 | Full address mode | |

**Bits 13 and 12—Reserved:** Can be read or written to.

**Bits 11 and 9—Data Transfer Acknowledge (DTA):** These bits enable or disable clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.

When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU or DTC.

When the DTE = 1 and the DTA = 0, the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU or DTC in parallel. In this case, the interrupt source should be cleared by the CPU or DTC transfer.

When the DTE = 0, the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU or DTC regardless of the DTA bit setting.

The state of the DTME bit does not affect the above operations.

**Bit 11—Data Transfer Acknowledge 1 (DTA1):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1 data transfer factor setting.

**Bit 11**

| DTA1 | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bit 9—Data Transfer Acknowledge 0 (DTA0):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0 data transfer factor setting.

**Bit 9**

| DTA0 | Description |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value) |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

**Bits 10 and 8—Reserved:** Can be read or written to.

**Bits 7 and 5—Data Transfer Master Enable (DTME):** Together with the DTE bit, these bits control enabling or disabling of data transfer on the relevant channel. When both the DTME bit and the DTE bit are set to 1, transfer is enabled for the channel.

If the relevant channel is in the middle of a burst mode transfer when an NMI interrupt is generated, the DTME bit is cleared, the transfer is interrupted, and bus mastership passes to the CPU. When the DTME bit is subsequently set to 1 again, the interrupted transfer is resumed. In block transfer mode, however, the DTME bit is not cleared by an NMI interrupt, and transfer is not interrupted.

The conditions for the DTME bit being cleared to 0 are as follows:

- When initialization is performed
- When NMI is input in burst mode
- When 0 is written to the DTME bit

The condition for DTME being set to 1 is as follows:

- When 1 is written to DTME after DTME is read as 0

**Bit 7—Data Transfer Master Enable 1 (DTME1):** Enables or disables data transfer on channel 1.

**Bit 7**

| DTME1 | Description | |
|---|---|---|
| 0 | Data transfer disabled. In burst mode, cleared to 0 by an NMI interrupt | (Initial value) |
| 1 | Data transfer enabled | |

**Bit 5—Data Transfer Master Enable 0 (DTME0):** Enables or disables data transfer on channel 0.

**Bit 5**

| DTME0 | Description | |
|---|---|---|
| 0 | Data transfer disabled. In normal mode, cleared to 0 by an NMI interrupt | (Initial value) |
| 1 | Data transfer enabled | |

**Bits 6 and 4—Data Transfer Enable (DTE):** When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU or DTC. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU.

The conditions for the DTE bit being cleared to 0 are as follows:

- When initialization is performed
- When the specified number of transfers have been completed
- When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason

When DTE = 1 and DTME = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed.

The condition for the DTE bit being set to 1 is as follows:

- When 1 is written to the DTE bit after the DTE bit is read as 0

**Bit 6—Data Transfer Enable 1 (DTE1):** Enables or disables data transfer on channel 1.

**Bit 6**

| DTE1 | Description | |
|---|---|---|
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bit 4—Data Transfer Enable 0 (DTE0):** Enables or disables data transfer on channel 0.

**Bit 4**

| DTE0 | Description | |
|---|---|---|
| 0 | Data transfer disabled | (Initial value) |
| 1 | Data transfer enabled | |

**Bits 3 and 1—Data Transfer Interrupt Enable B (DTIEB):** These bits enable or disable an interrupt to the CPU or DTC when transfer is interrupted. If the DTIEB bit is set to 1 when DTME = 0, the DMAC regards this as indicating a break in the transfer, and issues a transfer break interrupt request to the CPU or DTC.

A transfer break interrupt can be canceled either by clearing the DTIEB bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTME bit to 1.

**Bit 3—Data Transfer Interrupt Enable 1B (DTIE1B):** Enables or disables the channel 1 transfer break interrupt.

**Bit 3**

| DTIE1B | Description | |
|---|---|---|
| 0 | Transfer break interrupt disabled | (Initial value) |
| 1 | Transfer break interrupt enabled | |

**Bit 1—Data Transfer Interrupt Enable 0B (DTIE0B):** Enables or disables the channel 0 transfer break interrupt.

**Bit 1**

| DTIE0B | Description | |
|---|---|---|
| 0 | Transfer break interrupt disabled | (Initial value) |
| 1 | Transfer break interrupt enabled | |

**Bits 2 and 0—Data Transfer End Interrupt Enable A (DTIEA):** These bits enable or disable an interrupt to the CPU or DTC when transfer ends. If DTIEA bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

A transfer end interrupt can be canceled either by clearing the DTIEA bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the transfer counter and address register again, and then setting the DTE bit to 1.

**Bit 2—Data Transfer Interrupt Enable 1A (DTIE1A):** Enables or disables the channel 1 transfer end interrupt.

**Bit 2**

| DTIE1A | Description | |
|---|---|---|
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

**Bit 0—Data Transfer Interrupt Enable 0A (DTIE0A):** Enables or disables the channel 0 transfer end interrupt.

**Bit 0**

| DTIE0A | Description | |
|---|---|---|
| 0 | Transfer end interrupt disabled | (Initial value) |
| 1 | Transfer end interrupt enabled | |

## 7.4 Register Descriptions (3)

### 7.4.1 DMA Write Enable Register (DMAWER)

The DMAC can activate the DTC with a transfer end interrupt, rewrite the channel on which the transfer ended using a DTC chain transfer, and reactivate the DTC. DMAWER applies restrictions so that specific bits of DMACR for the specific channel, and also DMATCR and DMABCR, can be changed to prevent inadvertent rewriting of registers other than those for the channel concerned, The restrictions applied by DMAWER are valid for the DTC.

Figure 7-2 shows the transfer areas for activating the DTC with a channel 0A transfer end interrupt, and reactivating channel 0A. The address register and count register area is re-set by the first DTC transfer, then the control register area is re-set by the second DTC chain transfer.

When re-setting the control register area, perform masking by setting bits in DMAWER to prevent modification of the contents of the other channels.

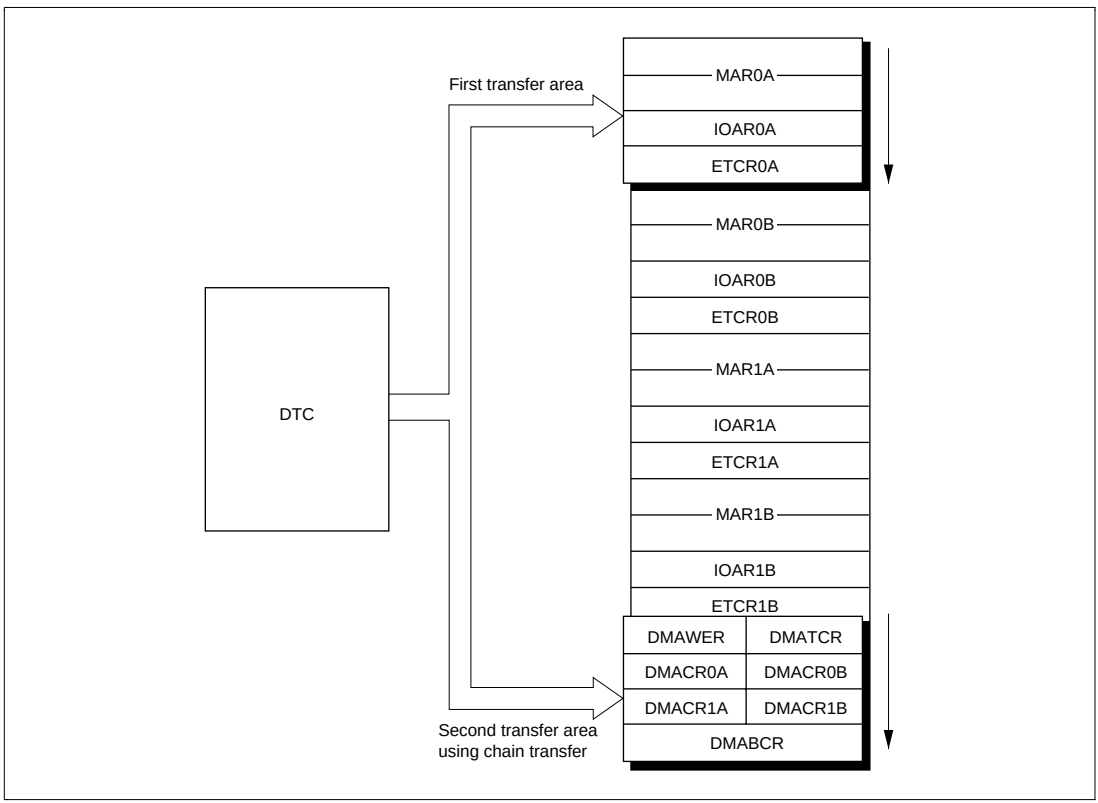



**Figure 7-2 Areas for Register Re-Setting by DTC (Example: Channel 0A)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMAWER | : | — | — | — | — | WE1B | WE1A | WE0B | WE0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

DMAWER is an 8-bit readable/writable register that controls enabling or disabling of writes to the DMACR, DMABCR, and DMATCR by the DTC.

DMAWER is initialized to H'00 by a reset, and in standby mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Write Enable 1B (WE1B):** Enables or disables writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR by the DTC.

**Bit 3**

| WE1B | Description |
|---|---|
| 0 | Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are disabled (Initial value) |
| 1 | Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are enabled |

**Bit 2—Write Enable 1A (WE1A):** Enables or disables writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR by the DTC.

**Bit 2**

| WE1A | Description |
|---|---|
| 0 | Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are disabled (Initial value) |
| 1 | Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are enabled |

**Bit 1—Write Enable 0B (WE0B):** Enables or disables writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR.

**Bit 1**

| WE0B | Description |
|---|---|
| 0 | Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are disabled (Initial value) |
| 1 | Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are enabled |

**Bit 0—Write Enable 0A (WE0A):** Enables or disables writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR.

**Bit 0**

| WE0A | Description |
|---|---|
| 0 | Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are disabled (Initial value) |
| 1 | Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are enabled |

Writes by the DTC to bits 15 to 12 (FAE and SAE) in DMABCR are invalid regardless of the DMAWER settings. These bits should be changed, if necessary, by CPU processing.

In writes by the DTC to bits 7 to 4 (DTE) in DMABCR, 1 can be written without first reading 0. To reactivate a channel set to full address mode, write 1 to both Write Enable A and Write Enable B for the channel to be reactivated.

MAR, IOAR, and ETCR are always write-enabled regardless of the DMAWER settings. When modifying these registers, the channel for which the modification is to be made should be halted.

### 7.4.2 DMA Terminal Control Register (DMATCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMATCR | : | — | — | TEE1 | TEE0 | — | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | — | — | — | — |

DMATCR is an 8-bit readable/writable register that controls enabling or disabling of DMAC transfer end pin output. A port can be set for output automatically, and a transfer end signal output, by setting the appropriate bit.

DMATCR is initialized to H'00 by a reset, and in standby mode.

**Bits 7 and 6—Reserved:** Read-only bits, always read as 0.

**Bit 5—Transfer End Enable 1 (TEE1):** Enables or disables transfer end pin 1 ($\overline{\text{TEND}}_1$) output.

**Bit 5**

| TEE1 | Description | |
|---|---|---|
| 0 | $\overline{\text{TEND}}_1$ pin output disabled | (Initial value) |
| 1 | $\overline{\text{TEND}}_1$ pin output enabled | |

**Bit 4—Transfer End Enable 0 (TEE0):** Enables or disables transfer end pin 0 ($\overline{\text{TEND}}_0$) output.

**Bit 4**

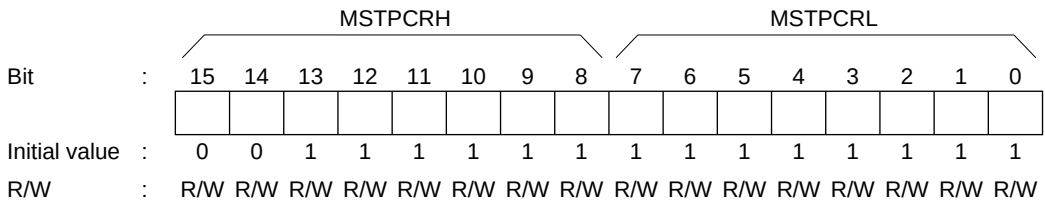
| TEE0 | Description | |
|---|---|---|
| 0 | $\overline{\text{TEND}}_0$ pin output disabled | (Initial value) |
| 1 | $\overline{\text{TEND}}_0$ pin output enabled | |

The $\overline{\text{TEND}}$ pins are assigned only to channel B in short address mode.

The transfer end signal indicates the transfer cycle in which the transfer counter reached 0, regardless of the transfer source. An exception is block transfer mode, in which the transfer end signal indicates the transfer cycle in which the block counter reached 0.

**Bits 3 to 0—Reserved:** Read-only bits, always read as 0.

### 7.4.3 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP15 bit in MSTPCR is set to 1, the DMAC operation stops at the end of the bus cycle and a transition is made to module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 15—Module Stop (MSTP15):** Specifies the DMAC module stop mode.

**Bits 15**

| MSTP15 | Description | |
|---|---|---|
| 0 | DMAC module stop mode cleared | (Initial value) |
| 1 | DMAC module stop mode set | |

## 7.5 Operation

### 7.5.1 Transfer Modes

Table 7-5 lists the DMAC modes.

**Table 7-5 DMAC Transfer Modes**

| Transfer Mode | | | Transfer Source | Remarks |
|---|---|---|---|---|
| Short address mode | Dual address mode | (1) Sequential mode<br>(2) Idle mode<br>(3) repeat mode | • TPU channel 0 to 5 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• A/D converter conversion end interrupt<br>• External request | • Up to 4 channels can operate independently<br>• External request applies to channel B only<br>• Single address mode applies to channel B only<br>• Modes (1), (2), and (3) can also be specified for single address mode |
| | (4) Single address mode | | | |
| Full address mode | | (5) Normal mode | • External request<br>• Auto-request | • Max. 2-channel operation, combining channels A and B |
| | | (6) Block transfer mode | • TPU channel 0 to 5 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• A/D converter conversion end interrupt<br>• External request | • With auto-request, burst mode transfer or cycle steal transfer can be selected |

Operation in each mode is summarized below.

**(1) Sequential mode**

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.

**(2) Idle mode**

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer source address and transfer destination address are fixed. The transfer direction is programmable.

**(3) Repeat mode**

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. When the specified number of transfers have been completed, the addresses and transfer counter are restored to their original settings, and operation is continued. No interrupt request is sent to the CPU or DTC. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.

**(4) Single address mode**

In response to a single transfer request, the specified number of transfers are carried out between external memory and an external device, one byte or one word at a time. Unlike dual address mode, source and destination accesses are performed in parallel. Therefore, either the source or the destination is an external device which can be accessed with a strobe alone, using the $\overline{\text{DACK}}$ pin. One address is specified as 24 bits, and for the other, the pin is set automatically. The transfer direction is programmable.

Modes (1), (2) and (3) can also be specified for single address mode.

**(5) Normal mode**

- Auto-request

  By means of register settings only, the DMAC is activated, and transfer continues until the specified number of transfers have been completed. An interrupt request can be sent to the CPU or DTC when transfer is completed. Both addresses are specified as 24 bits.

  — Cycle steal mode: The bus is released to another bus master every byte or word transfer.

  — Burst mode: The bus is held and transfer continued until the specified number of transfers have been completed.

- External request

  In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time.  An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. Both addresses are specified as 24 bits.

**(6) Block transfer mode**

In response to a single transfer request, a block transfer of the specified block size is carried out. This is repeated the specified number of times, once each time there is a transfer request. At the end of each single block transfer, one address is restored to its original setting. An interrupt request can be sent to the CPU or DTC when the specified number of block transfers have been completed. Both addresses are specified as 24 bits.

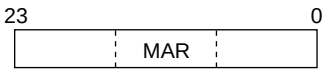
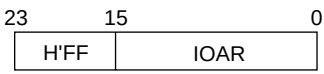
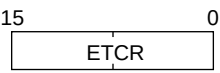
### 7.5.2 Sequential Mode

Sequential mode can be specified by clearing the RPE bit in DMACR to 0. In sequential mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 7-6 summarizes register functions in sequential mode.

**Table 7-6 Register Functions in Sequential Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
|  | Source address register | Destination address register | Start address of transfer destination or transfer source | Incremented/ decremented every transfer |
|  | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
|  | Transfer counter | | Number of transfers | Decremented every transfer; transfer ends when count reaches H'0000 |

**Legend**

MAR : Memory address register
IOAR : I/O address register
ETCR : Transfer count register
DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

Figure 7-3 illustrates operation in sequential mode.

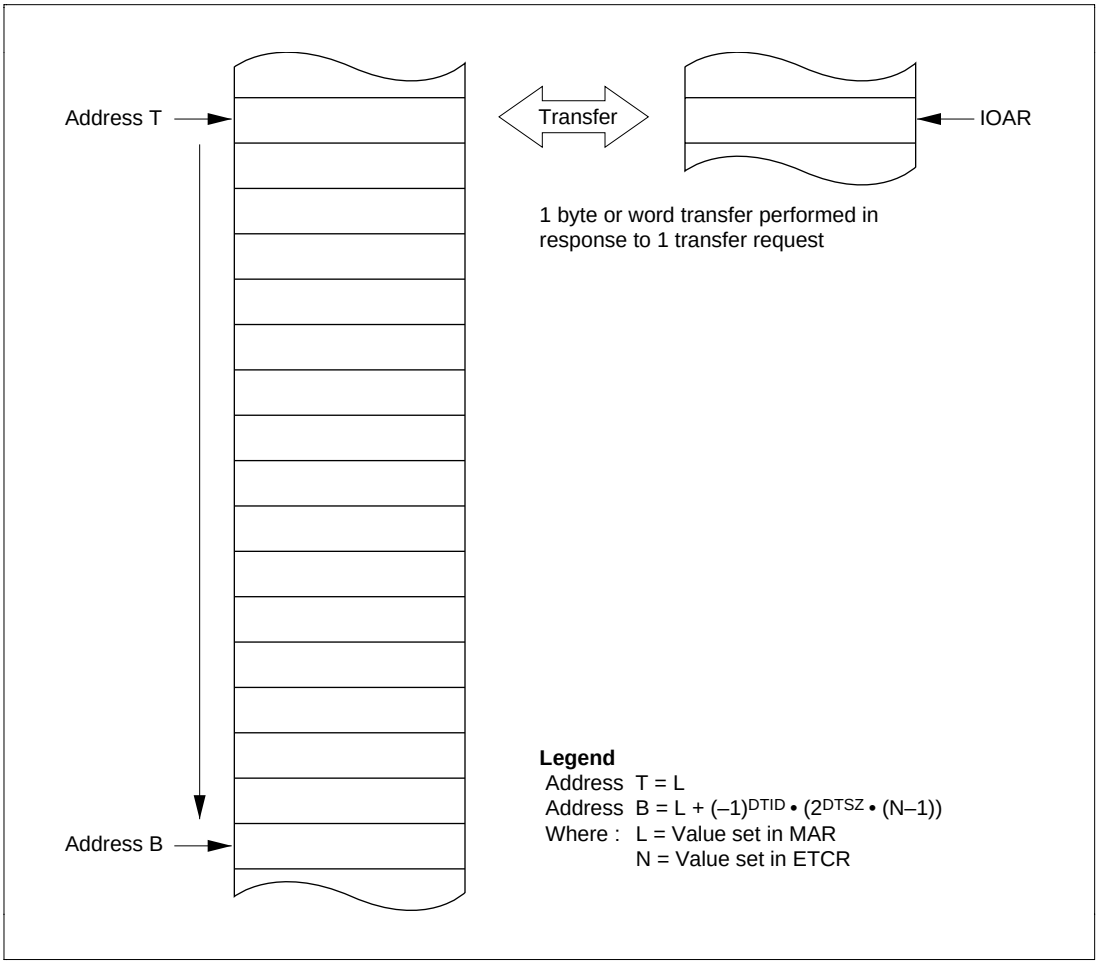


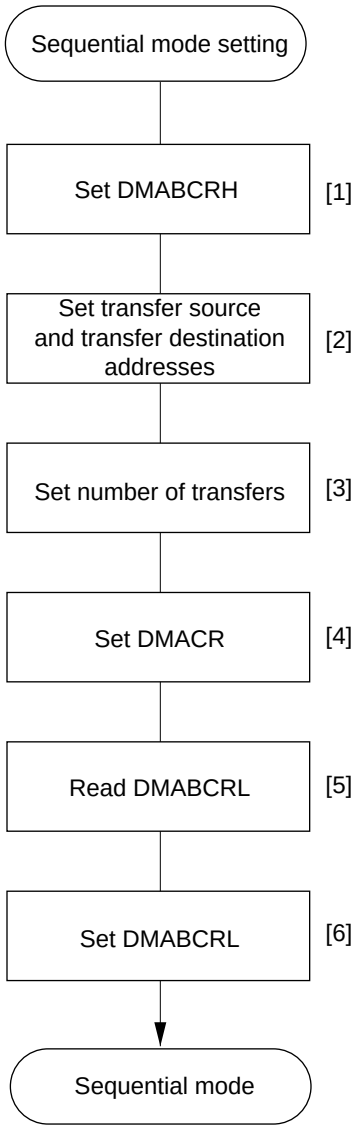

**Figure 7-3 Operation in Sequential Mode**

The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCR, is 65,536.

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

Figure 7-4 shows an example of the setting procedure for sequential mode.




[1] Set each bit in DMABCRH.
- Clear the FAE bit to 0 to select short address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address and transfer destination address in MAR and IOAR.

[3] Set the number of transfers in ETCR.

[4] Set each bit in DMACR.
- Set the transfer data size with the DTSZ bit.
- Specify whether MAR is to be incremented or decremented with the DTID bit.
- Clear the RPE bit to 0 to select sequential mode.
- Specify the transfer direction with the DTDIR bit.
- Select the activation source with bits DTF3 to DTF0.

[5] Read the DTE bit in DMABCRL as 0.

[6] Set each bit in DMABCRL.
- Specify enabling or disabling of transfer end interrupts with the DTIE bit.
- Set the DTE bit to 1 to enable transfer.

**Figure 7-4 Example of Sequential Mode Setting Procedure**

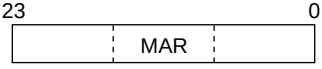
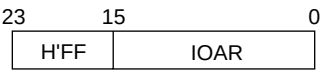
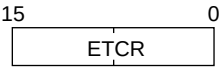
### 7.5.3 Idle Mode

Idle mode can be specified by setting the RPE bit and DTIE bit in DMACR to 1. In idle mode, one byte or word is transferred in response to a single transfer request, and this is executed the number of times specified in ETCR.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 7-7 summarizes register functions in idle mode.

**Table 7-7 Register Functions in Idle Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
|  | Source address register | Destination address register | Start address of transfer destination or transfer source | Fixed |
|  | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
|  | Transfer counter | | Number of transfers | Decremented every transfer; transfer ends when count reaches H'0000 |

**Legend**

MAR : Memory address register
IOAR : I/O address register
ETCR : Transfer count register
DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is neither incremented nor decremented each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

Figure 7-5 illustrates operation in idle mode.

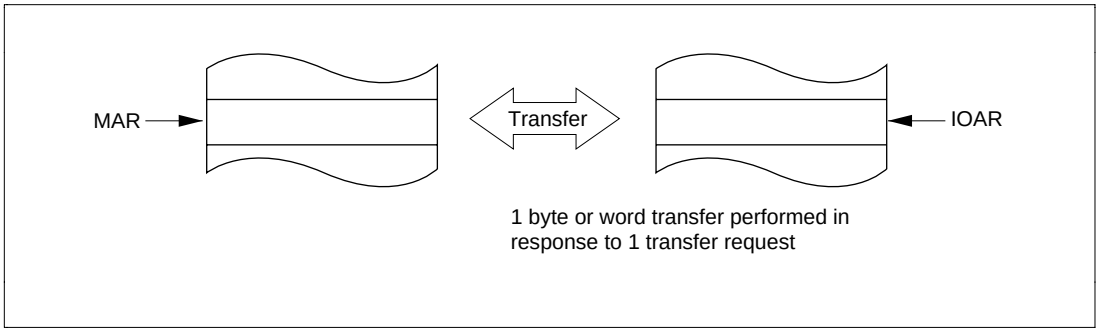



**Figure 7-5 Operation in Idle Mode**

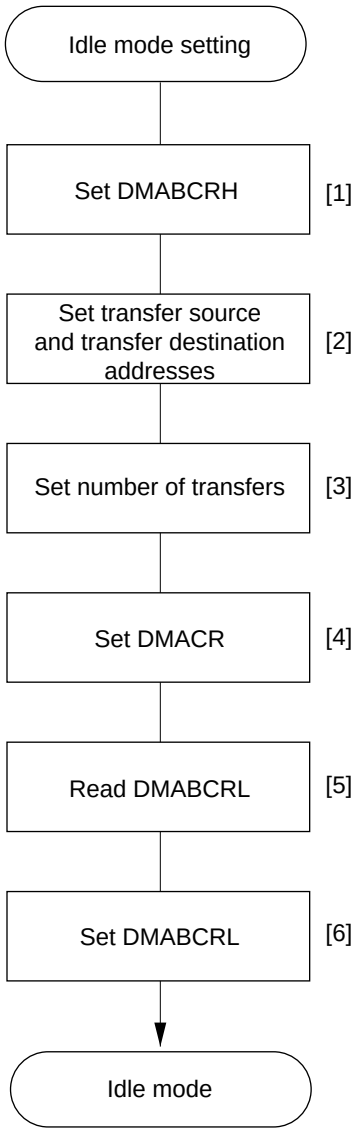
The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCR, is 65,536.

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

When the DMAC is used in single address mode, only channel B can be set.

Figure 7-6 shows an example of the setting procedure for idle mode.




[1] Set each bit in DMABCRH.
- Clear the FAE bit to 0 to select short address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address and transfer destination address in MAR and IOAR.

[3] Set the number of transfers in ETCR.

[4] Set each bit in DMACR.
- Set the transfer data size with the DTSZ bit.
- Specify whether MAR is to be incremented or decremented with the DTID bit.
- Set the RPE bit to 1.
- Specify the transfer direction with the DTDIR bit.
- Select the activation source with bits DTF3 to DTF0.

[5] Read the DTE bit in DMABCRL as 0.

[6] Set each bit in DMABCRL.
- Set the DTIE bit to 1.
- Set the DTE bit to 1 to enable transfer.

**Figure 7-6 Example of Idle Mode Setting Procedure**

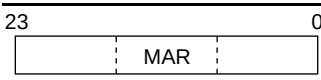
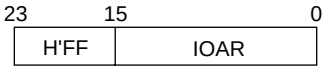
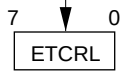
### 7.5.4 Repeat Mode

Repeat mode can be specified by setting the RPE bit in DMACR to 1, and clearing the DTIE bit to 0. In repeat mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR. On completion of the specified number of transfers, MAR and ETCRL are automatically restored to their original settings and operation continues.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 7-8 summarizes register functions in repeat mode.

**Table 7-8 Register Functions in Repeat Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
|  | Source address register | Destination address register | Start address of transfer destination or transfer source | Incremented/ decremented every transfer. Initial setting is restored when value reaches H'0000 |
|  | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
| | Holds number of transfers | | Number of transfers | Fixed |
|  | Transfer counter | | Number of transfers | Decremented every transfer. Loaded with ETCRH value when count reaches H'00 |

**Legend**

MAR : Memory address register
IOAR : I/O address register
ETCR : Transfer count register
DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

The number of transfers is specified as 8 bits by ETCRH and ETCRL. The maximum number of transfers, when H'00 is set in both ETCRH and ETCRL, is 256.

In repeat mode, ETCRL functions as the transfer counter, and ETCRH is used to hold the number of transfers. ETCRL is decremented by 1 each time a transfer is executed, and when its value reaches H'00, it is loaded with the value in ETCRH. At the same time, the value set in MAR is restored in accordance with the values of the DTSZ and DTID bits in DMACR. The MAR restoration operation is as shown below.

$$MAR = MAR - (-1)^{DTID \cdot 2DTSZ} \cdot ETCRH$$

The same value should be set in ETCRH and ETCRL.

In repeat mode, operation continues until the DTE bit is cleared. To end the transfer operation, therefore, you should clear the DTE bit to 0. A transfer end interrupt request is not sent to the CPU or DTC.

By setting the DTE bit to 1 again after it has been cleared, the operation can be restarted from the transfer after that terminated when the DTE bit was cleared.

Figure 7-7 illustrates operation in repeat mode.

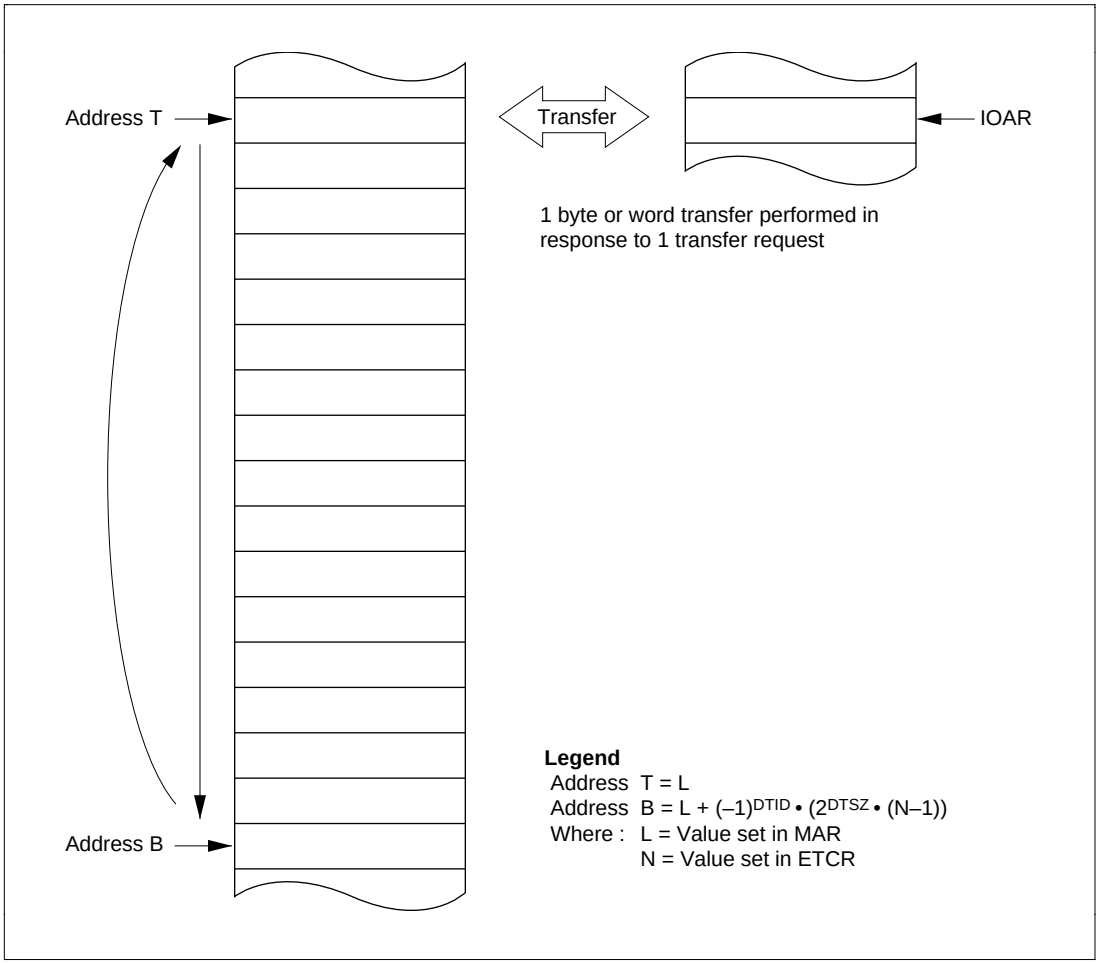
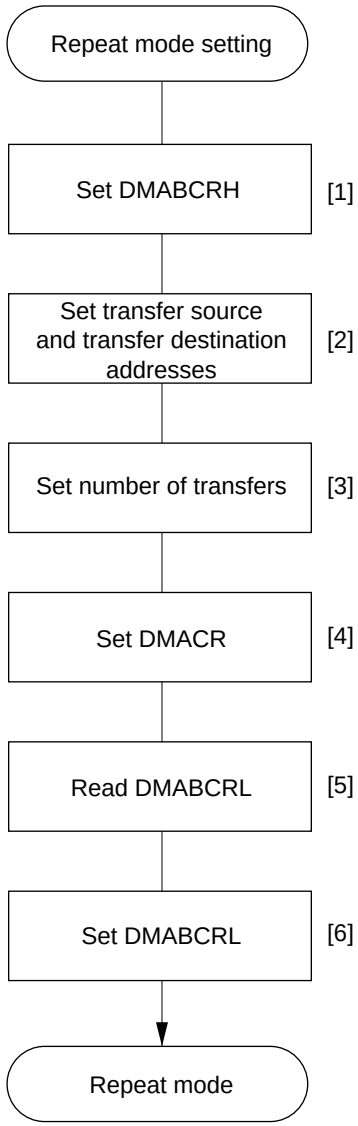



**Figure 7-7 Operation in Repeat mode**

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

Figure 7-8 shows an example of the setting procedure for repeat mode.




[1] Set each bit in DMABCRH.
- Clear the FAE bit to 0 to select short address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address and transfer destination address in MAR and IOAR.

[3] Set the number of transfers in both ETCRH and ETCRL.

[4] Set each bit in DMACR.
- Set the transfer data size with the DTSZ bit.
- Specify whether MAR is to be incremented or decremented with the DTID bit.
- Set the RPE bit to 1.
- Specify the transfer direction with the DTDIR bit.
- Select the activation source with bits DTF3 to DTF0.

[5] Read the DTE bit in DMABCRL as 0.

[6] Set each bit in DMABCRL.
- Clear the DTIE bit to 0.
- Set the DTE bit to 1 to enable transfer.

**Figure 7-8 Example of Repeat Mode Setting Procedure**

### 7.5.5 Single Address Mode

Single address mode can only be specified for channel B. This mode can be specified by setting the SAE bit in DMABCR to 1 in short address mode.

One address is specified by MAR, and the other is set automatically to the data transfer acknowledge pin ($\overline{\text{DACK}}$). The transfer direction can be specified by the DTDIR in DMACR.

Table 7-9 summarizes register functions in single address mode.

**Table 7-9 Register Functions in Single Address Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
| 23 MAR 0 | Source address register | Destination address register | Start address of transfer destination or transfer source | * |
| $\overline{\text{DACK}}$ pin | Write strobe | Read strobe | (Set automatically by SAE bit; IOAR is invalid) | Strobe for external device |
| 15 ETCR 0 | Transfer counter | | Number of transfers | * |

**Legend**

MAR : Memory address register
IOAR : I/O address register
ETCR : Transfer count register
DTDIR : Data transfer direction bit
$\overline{\text{DACK}}$ : Data transfer acknowledge

Note: * See the operation descriptions in sections 7.5.2, Sequential Mode, 7.5.3, Idle Mode, and 7.5.4, Repeat Mode.

MAR specifies the start address of the transfer source or transfer destination as 24 bits.

IOAR is invalid; in its place the strobe for external devices ($\overline{\text{DACK}}$) is output.

Figure 7-9 illustrates operation in single address mode (when sequential mode is specified).

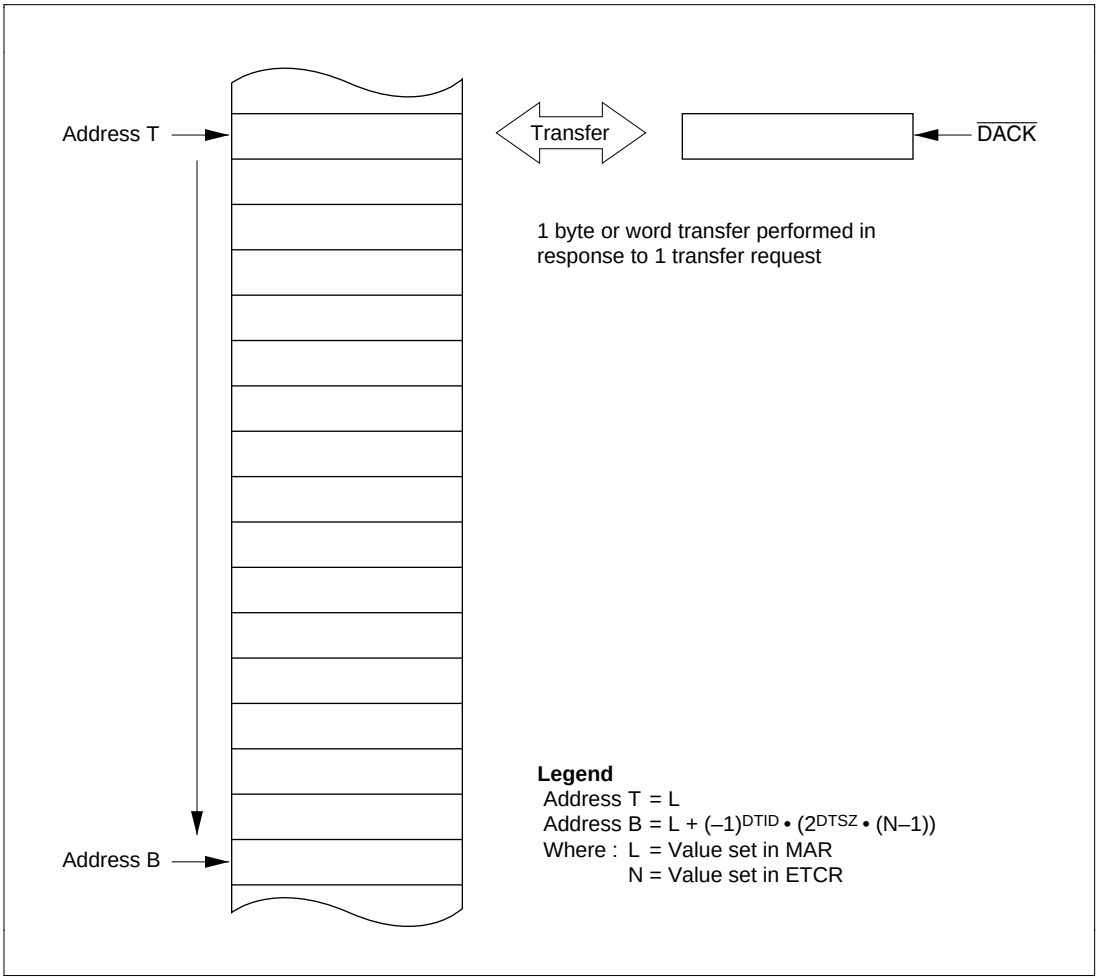



**Figure 7-9 Operation in Single Address Mode (When Sequential Mode is Specified)**

Figure 7-10 shows an example of the setting procedure for single address mode (when sequential mode is specified).

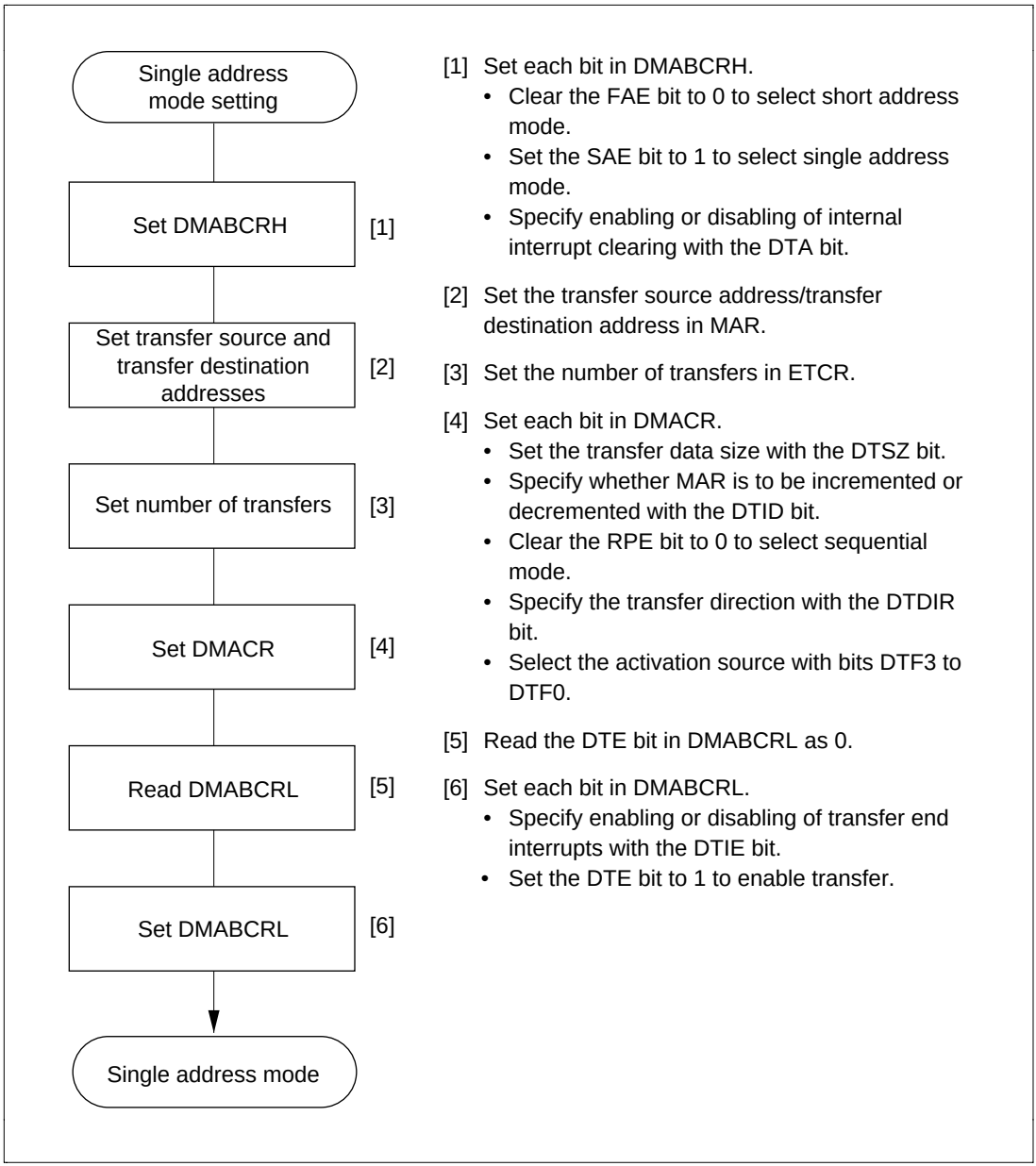



**Figure 7-10 Example of Single Address Mode Setting Procedure (When Sequential Mode is Specified)**

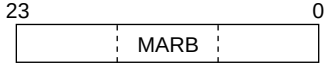
### 7.5.6 Normal Mode

In normal mode, transfer is performed with channels A and B used in combination. Normal mode can be specified by setting the FAE bit in DMABCR to 1 and clearing the BLKE bit in DMACRA to 0.

In normal mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCRA. The transfer source is specified by MARA, and the transfer destination by MARB.

Table 7-10 summarizes register functions in normal mode.

**Table 7-10 Register Functions in Normal Mode**

| Register | Function | Initial Setting | Operation |
|---|---|---|---|
| | Source address register | Start address of transfer source | Incremented/decremented every transfer, or fixed |
|  | Destination address register | Start address of transfer destination | Incremented/decremented every transfer, or fixed |
| | Transfer counter | Number of transfers | Decremented every transfer; transfer ends when count reaches H'0000 |

**Legend**

MARA : Memory address register A
MARB : Memory address register B
ETCRA : Transfer count register A

MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed.

Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB.

The number of transfers is specified by ETCRA as 16 bits. ETCRA is decremented each time a transfer is performed, and when its value reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCRA, is 65,536.

Figure 7-11 illustrates operation in normal mode.

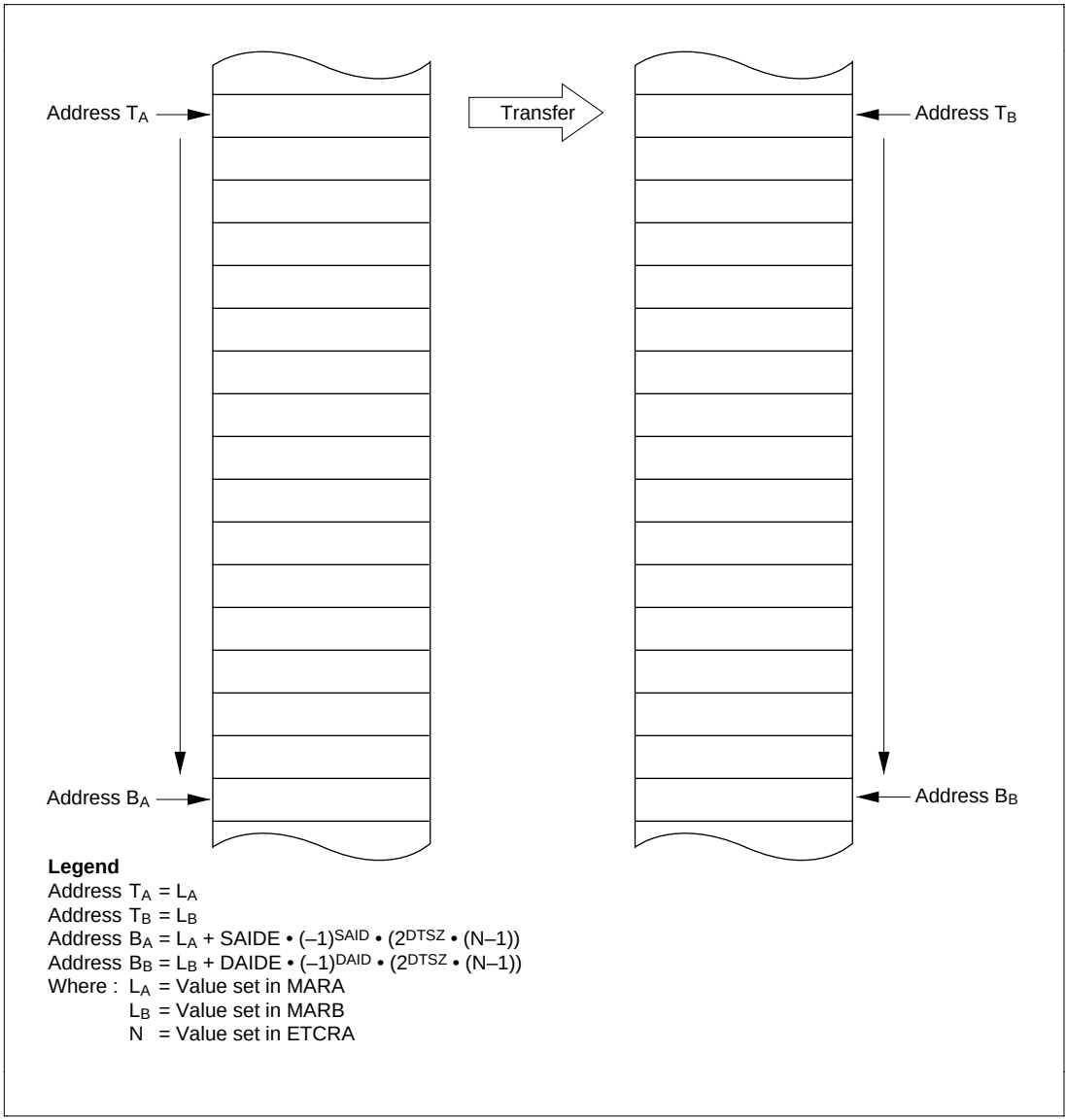



**Figure 7-11 Operation in Normal Mode**

Transfer requests (activation sources) are external requests and auto-requests.

With auto-request, the DMAC is only activated by register setting, and the specified number of transfers are performed automatically. With auto-request, cycle steal mode or burst mode can be selected. In cycle steal mode, the bus is released to another bus master each time a transfer is performed. In burst mode, the bus is held continuously until transfer ends.

For setting details, see section 7.3.4, DMA Controller Register (DMACR).

Figure 7-12 shows an example of the setting procedure for normal mode.

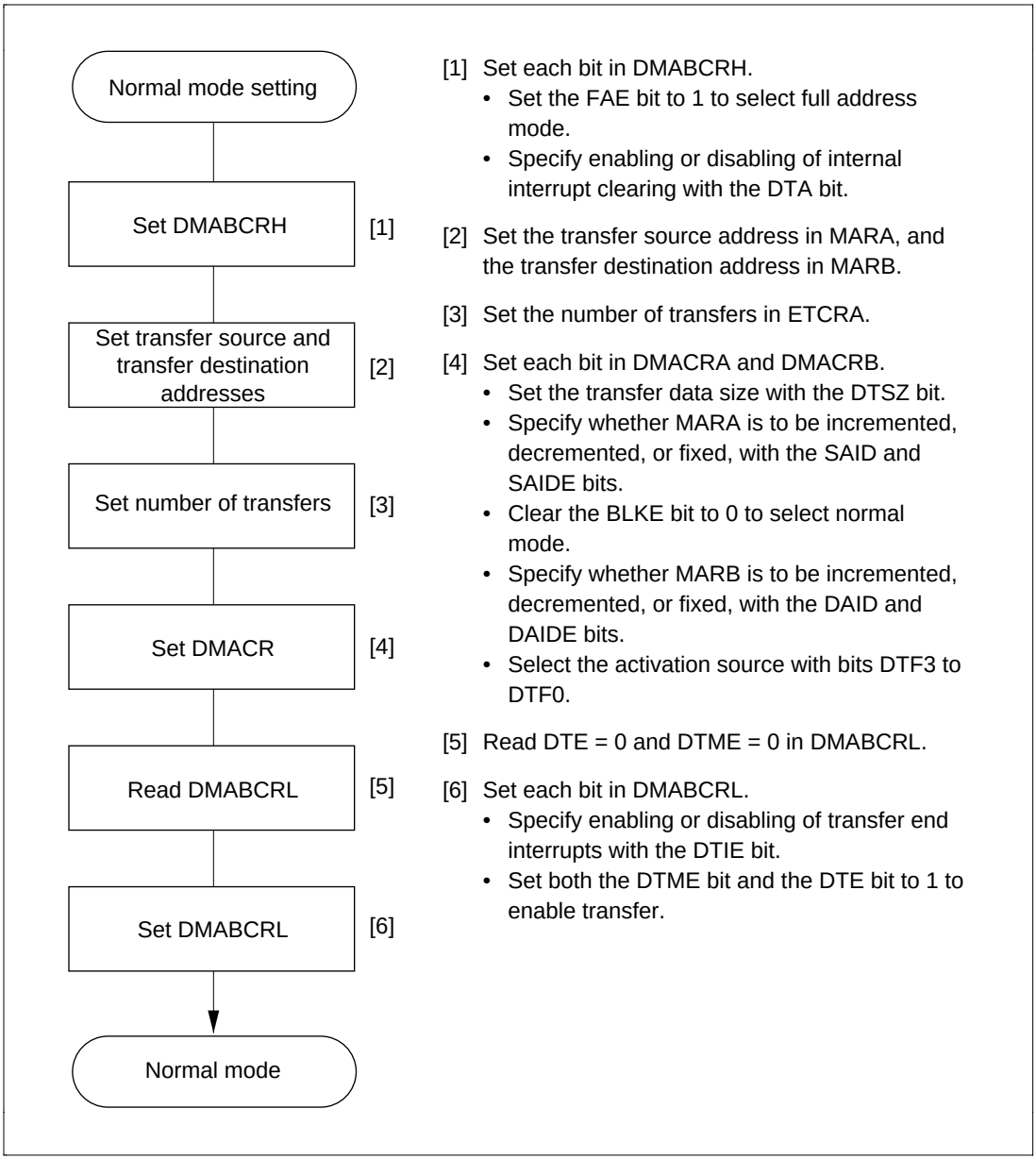



**Figure 7-12 Example of Normal Mode Setting Procedure**

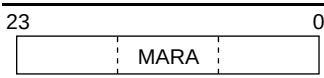
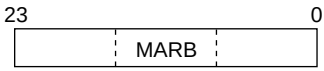
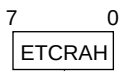
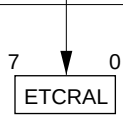
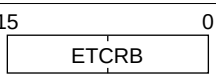
### 7.5.7 Block Transfer Mode

In block transfer mode, transfer is performed with channels A and B used in combination. Block transfer mode can be specified by setting the FAE bit in DMABCR and the BLKE bit in DMACRA to 1.

In block transfer mode, a transfer of the specified block size is carried out in response to a single transfer request, and this is executed the specified number of times. The transfer source is specified by MARA, and the transfer destination by MARB. Either the transfer source or the transfer destination can be selected as a block area (an area composed of a number of bytes or words).

Table 7-11 summarizes register functions in block transfer mode.

**Table 7-11 Register Functions in Block Transfer Mode**

| Register | Function | Initial Setting | Operation |
|---|---|---|---|
|  | Source address register | Start address of transfer source | Incremented/decremented every transfer, or fixed |
|  | Destination address register | Start address of transfer destination | Incremented/decremented every transfer, or fixed |
|  | Holds block size | Block size | Fixed |
|  | Block size counter | Block size | Decremented every transfer; ETCRH value copied when count reaches H'00 |
|  | Block transfer counter | Number of block transfers | Decremented every block transfer; transfer ends when count reaches H'0000 |

**Legend**

MARA : Memory address register A
MARB : Memory address register B
ETCRA : Transfer count register A
ETCRB : Transfer count register B

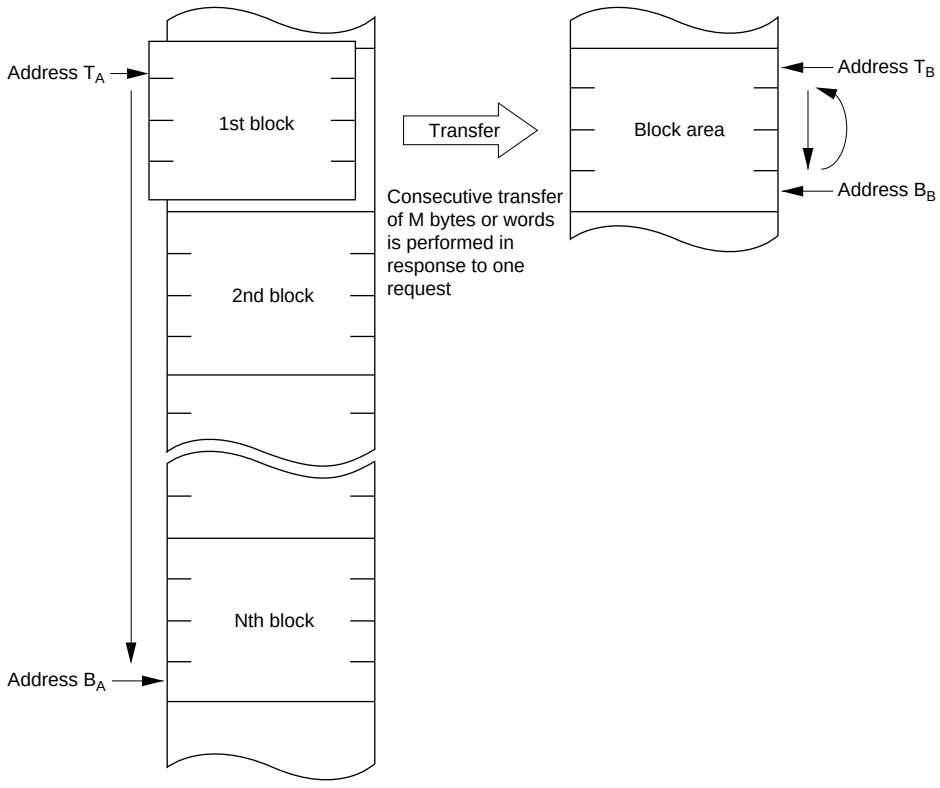
MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed.

Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB.

Whether a block is to be designated for MARA or for MARB is specified by the BLKDIR bit in DMACRA.

To specify the number of transfers, if M is the size of one block (where M = 1 to 256) and N transfers are to be performed (where N = 1 to 65,536), M is set in both ETCRAH and ETCRAL, and N in ETCRB.

Figure 7-13 illustrates operation in block transfer mode when MARB is designated as a block area.




**Legend**
Address $T_A$ = $L_A$
Address $T_B$ = $L_B$
Address $B_A$ = $L_A$ + SAIDE • $(-1)^{SAID}$ • $(2^{DTSZ}$ • (M•N–1))
Address $B_B$ = $L_B$ + DAIDE • $(-1)^{DAID}$ • $(2^{DTSZ}$ • (N–1))
Where : $L_A$ = Value set in MARA
$L_B$ = Value set in MARB
N = Value set in ETCRB
M = Value set in ETCRAH and ETCRAL

**Figure 7-13 Operation in Block Transfer Mode (BLKDIR = 0)**

Figure 7-14 illustrates operation in block transfer mode when MARA is designated as a block area.

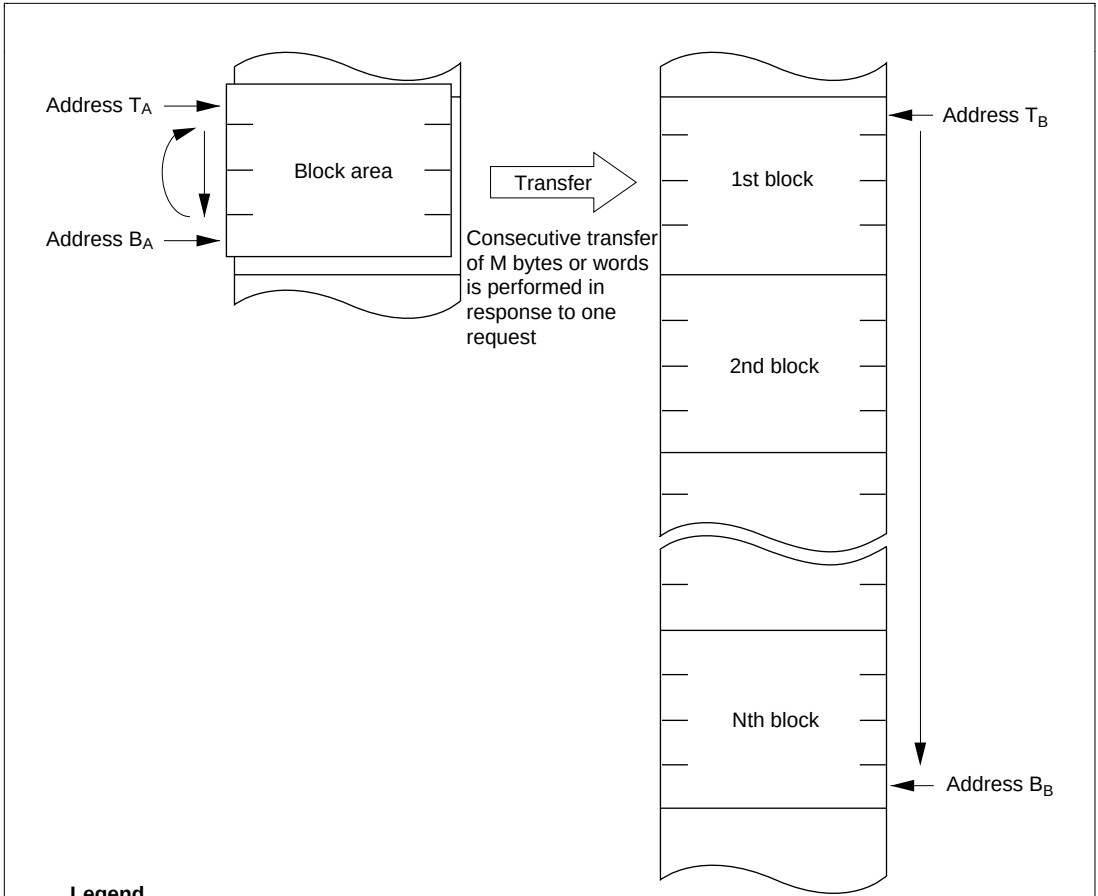



**Figure 7-14 Operation in Block Transfer Mode (BLKDIR = 1)**

ETCRAL is decremented by 1 each time a byte or word transfer is performed. In response to a single transfer request, burst transfer is performed until the value in ETCRAL reaches H'00. ETCRAL is then loaded with the value in ETCRAH. At this time, the value in the MAR register for which a block designation has been given by the BLKDIR bit in DMACRA is restored in accordance with the DTSZ, SAID/DAID, and SAIDE/DAIDE bits in DMACR.

ETCRB is decremented by 1 every block transfer, and when the count reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this point, an interrupt request is sent to the CPU or DTC.

Figure 7-15 shows the operation flow in block transfer mode.

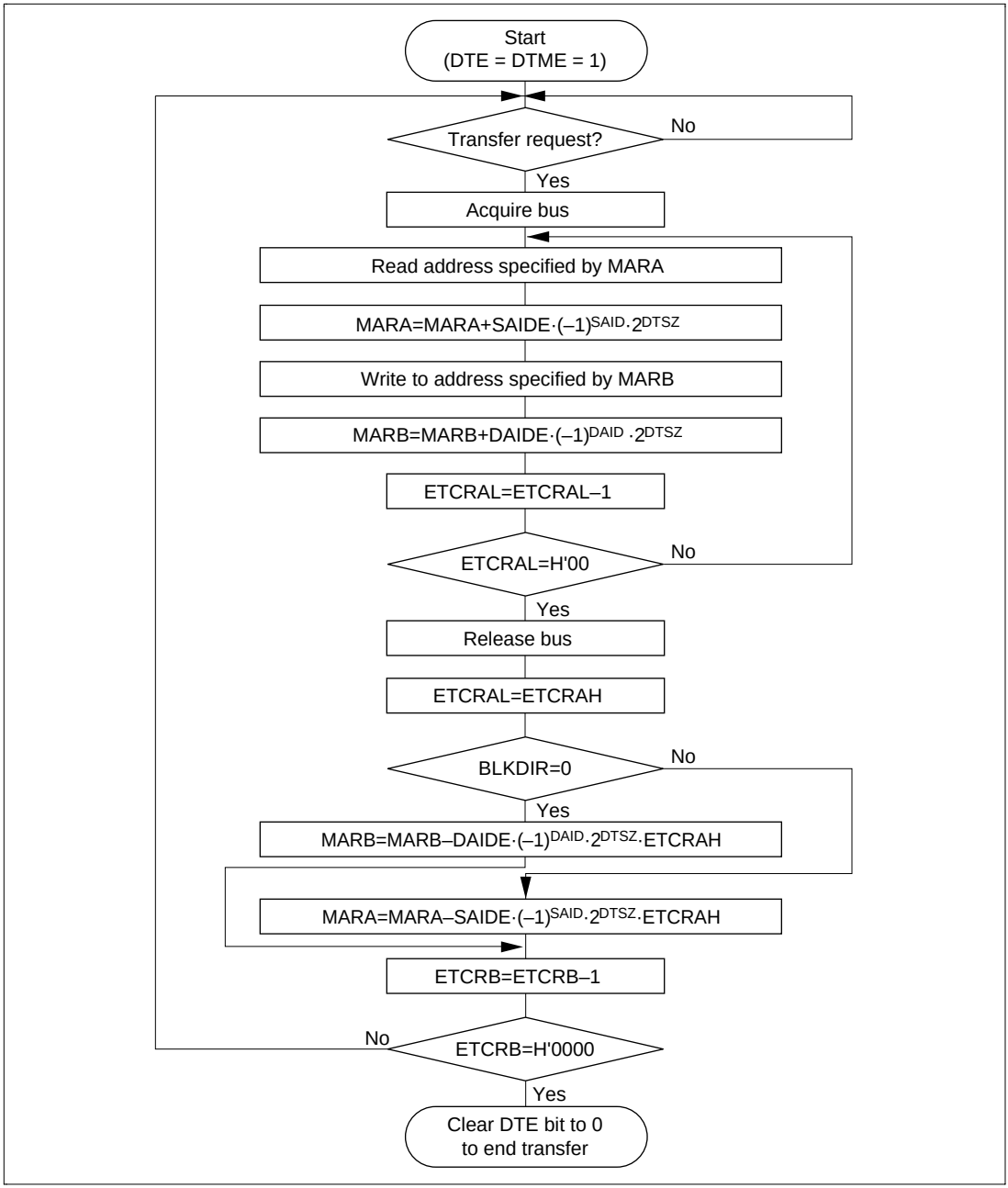
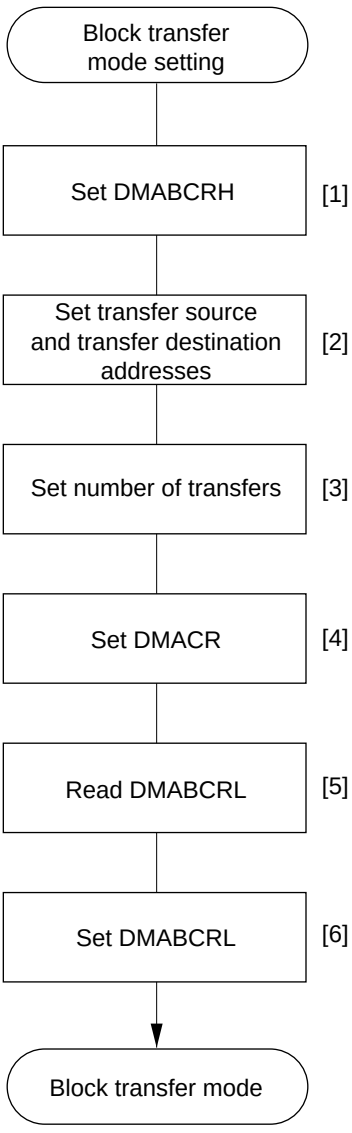



**Figure 7-15 Operation Flow in Block Transfer Mode**

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts.

For details, see section 7.3.4, DMA Control Register (DMACR).

Figure 7-16 shows an example of the setting procedure for block transfer mode.




[1] Set each bit in DMABCRH.
- Set the FAE bit to 1 to select full address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address in MARA, and the transfer destination address in MARB.

[3] Set the block size in both ETCRAH and ETCRAL. Set the number of transfers in ETCRB.

[4] Set each bit in DMACRA and DMACRB.
- Set the transfer data size with the DTSZ bit.
- Specify whether MARA is to be incremented, decremented, or fixed, with the SAID and SAIDE bits.
- Set the BLKE bit to 1 to select block transfer mode.
- Specify whether the transfer source or the transfer destination is a block area with the BLKDIR bit.
- Specify whether MARB is to be incremented, decremented, or fixed, with the DAID and DAIDE bits.
- Select the activation source with bits DTF3 to DTF0.

[5] Read DTE = 0 and DTME = 0 in DMABCRL.

[6] Set each bit in DMABCRL.
- Specify enabling or disabling of transfer end interrupts to the CPU with the DTIE bit.
- Set both the DTME bit and the DTE bit to 1 to enable transfer.

**Figure 7-16 Example of Block Transfer Mode Setting Procedure**

### 7.5.8 DMAC Activation Sources

DMAC activation sources consist of internal interrupts, external requests, and auto-requests. The activation sources that can be specified depend on the transfer mode and the channel, as shown in table 7-12.

**Table 7-12 DMAC Activation Sources**

| | | Short Address Mode | | Full Address Mode | |
|---|---|---|---|---|---|
| **Activation Source** | | **Channels 0A and 1A** | **Channels 0B and 1B** | **Normal Mode** | **Block Transfer Mode** |
| Internal Interrupts | ADI | ○ | ○ | X | ○ |
| | TXI0 | ○ | ○ | X | ○ |
| | RXI0 | ○ | ○ | X | ○ |
| | TXI1 | ○ | ○ | X | ○ |
| | RXI1 | ○ | ○ | X | ○ |
| | TGI0A | ○ | ○ | X | ○ |
| | TGI1A | ○ | ○ | X | ○ |
| | TGI2A | ○ | ○ | X | ○ |
| | TGI3A | ○ | ○ | X | ○ |
| | TGI4A | ○ | ○ | X | ○ |
| | TGI5A | ○ | ○ | X | ○ |
| External Requests | $\overline{\text{DREQ}}$ pin falling edge input | X | ○ | ○ | ○ |
| | $\overline{\text{DREQ}}$ pin low-level input | X | ○ | ○ | ○ |
| Auto-request | | X | X | ○ | X |

**Legend**

○: Can be specified

X : Cannot be specified

**Activation by Internal Interrupt:** An interrupt request selected as a DMAC activation source can be sent simultaneously to the CPU and DTC. For details, see section 5, Interrupt Controller.

With activation by an internal interrupt, the DMAC accepts the request independently of the interrupt controller. Consequently, interrupt controller priority settings are not accepted.

If the DMAC is activated by a CPU interrupt source or an interrupt source that is not used as a DTC activation source (DTA = 1), the interrupt source flag is cleared automatically by the DMA transfer. With ADI, TXI, and RXI interrupts, however, the interrupt source flag is not cleared unless the prescribed register is accessed in a DMA transfer. If the same interrupt is used as an

activation source for more than one channel, the interrupt request flag is cleared when the highest-priority channel is activated first. Transfer requests for other channels are held pending in the DMAC, and activation is carried out in order of priority.

When DTE = 0, such as after completion of a transfer, a request from the selected activation source is not sent to the DMAC, regardless of the DTA bit. In this case, the relevant interrupt request is sent to the CPU or DTC.

In case of overlap with a CPU interrupt source or DTC activation source (DTA = 0), the interrupt request flag is not cleared by the DMAC.

**Activation by External Request:** If an external request ($\overline{\text{DREQ}}$ pin) is specified as an activation source, the relevant port should be set to input mode in advance.

Level sensing or edge sensing can be used for external requests.

External request operation in normal mode (short address mode or full address mode) is described below.

When edge sensing is selected, a 1-byte or 1-word transfer is executed each time a high-to-low transition is detected on the $\overline{\text{DREQ}}$ pin. The next transfer may not be performed if the next edge is input before transfer is completed.

When level sensing is selected, the DMAC stands by for a transfer request while the $\overline{\text{DREQ}}$ pin is held high. While the $\overline{\text{DREQ}}$ pin is held low, transfers continue in succession, with the bus being released each time a byte or word is transferred. If the $\overline{\text{DREQ}}$ pin goes high in the middle of a transfer, the transfer is interrupted and the DMAC stands by for a transfer request.

**Activation by Auto-Request:** Auto-request activation is performed by register setting only, and transfer continues to the end.

With auto-request activation, cycle steal mode or burst mode can be selected.

In cycle steal mode, the DMAC releases the bus to another bus master each time a byte or word is transferred. DMA and CPU cycles usually alternate.

In burst mode, the DMAC keeps possession of the bus until the end of the transfer, and transfer is performed continuously.

**Single Address Mode:** The DMAC can operate in dual address mode in which read cycles and write cycles are separate cycles, or single address mode in which read and write cycles are executed in parallel.

In dual address mode, transfer is performed with the source address and destination address specified separately.

In single address mode, on the other hand, transfer is performed between external space in which either the transfer source or the transfer destination is specified by an address, and an external device for which selection is performed by means of the $\overline{\text{DACK}}$ strobe, without regard to the address. Figure 7-16 shows the data bus in single address mode.

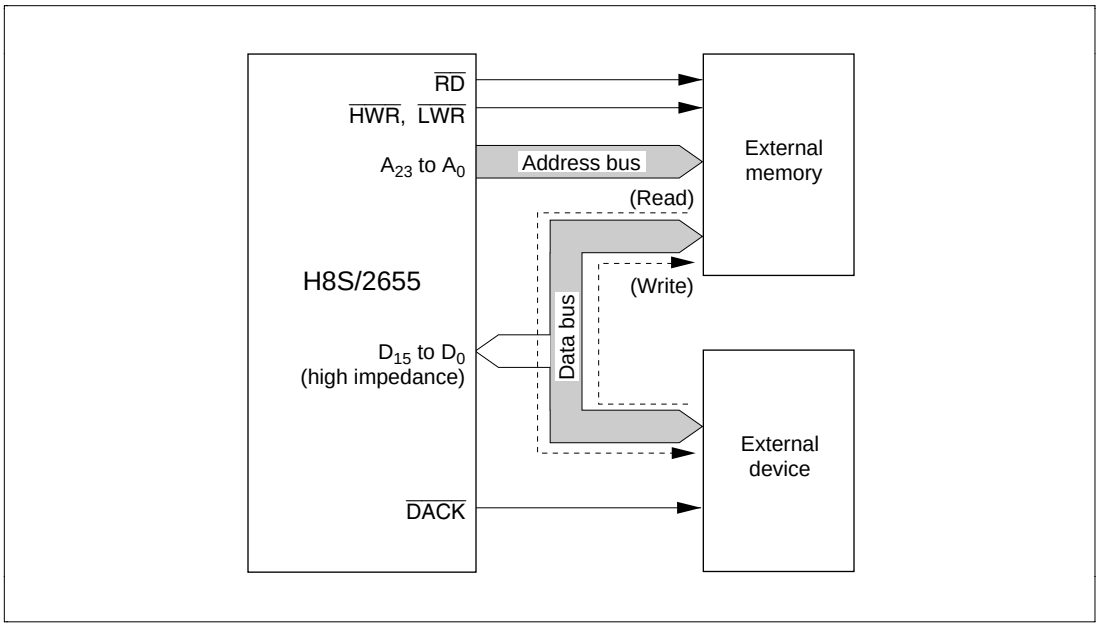



**Figure 7-17 Data Bus in Single Address Mode**

When using the DMAC for single address mode reading, transfer is performed from external memory to the external device, and the $\overline{\text{DACK}}$ pin functions as a write strobe for the external device. When using the DMAC for single address mode writing, transfer is performed from the external device to external memory, and the $\overline{\text{DACK}}$ pin functions as a read strobe for the external device. Since there is no directional control for the external device, one or other of the above single directions should be used.

Bus cycles in single address mode are in accordance with the settings of the bus controller for the external memory area. On the external device side, $\overline{\text{DACK}}$ is output in synchronization with the address strobe. For details of bus cycles, see section 7.5.11, DMAC Bus Cycles (Single Address Mode).

Do not specify internal space for transfer addresses in single address mode.

### 7.5.9 Basic DMAC Bus Cycles

An example of the basic DMAC bus cycle timing is shown in figure 7-18. In this example, word-size transfer is performed from 16-bit , 2-state access space to 8-bit, 3-state access space. When the bus is transferred from the CPU to the DMAC, a source address read and destination address write are performed. The bus is not released in response to another bus request, etc., between these read and write operations. As with CPU cycles, DMA cycles conform to the bus controller settings.

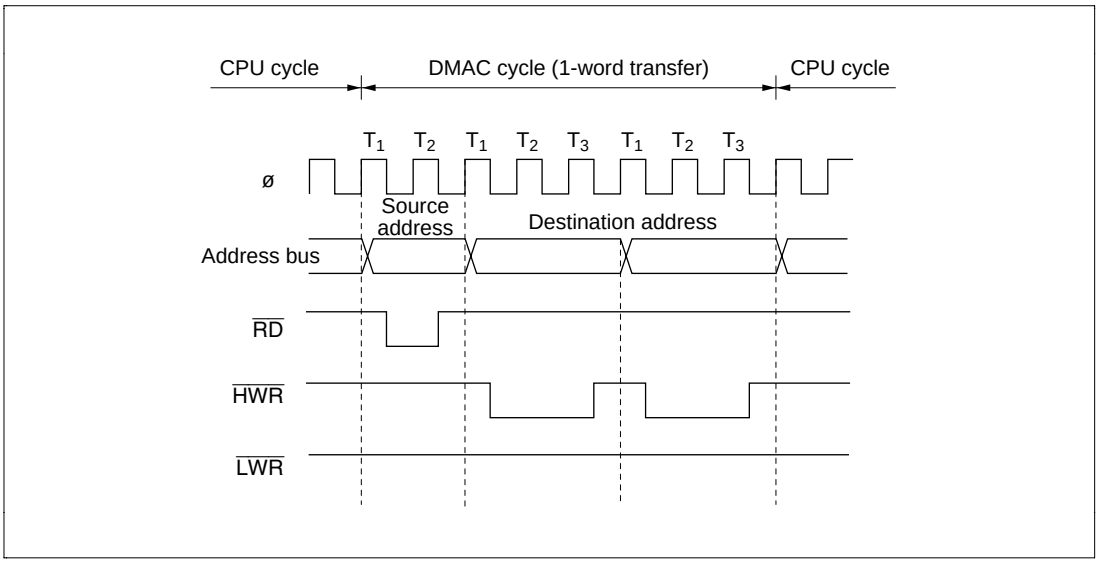



**Figure 7-18 Example of DMA Transfer Bus Timing**

The address is not output to the external address bus in an access to on-chip memory or an internal I/O register.

### 7.5.10 DMAC Bus Cycles (Dual Address Mode)

**Short Address Mode:** Figure 7-19 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and byte-size short address mode transfer (sequential/idle/repeat mode) is performed from external 8-bit, 2-state access space to internal I/O space.

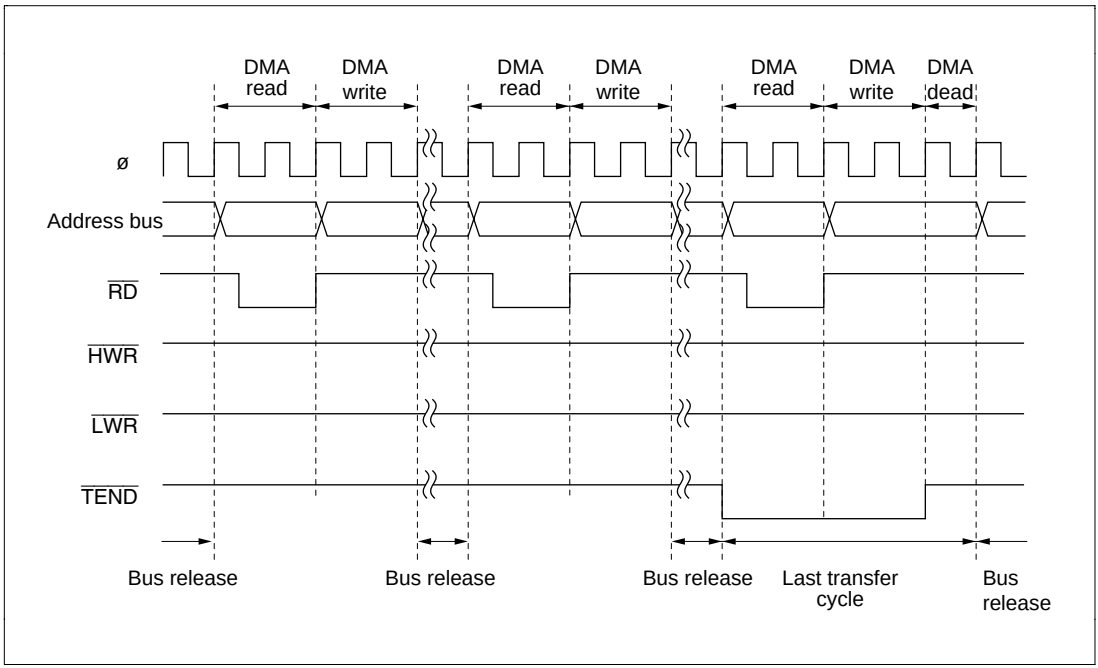



**Figure 7-19 Example of Short Address Mode Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released one or more bus cycles are inserted by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

In repeat mode, when $\overline{\text{TEND}}$ output is enabled, $\overline{\text{TEND}}$ output goes low in the transfer cycle in which the transfer counter reaches 0.

**Full Address Mode (Cycle Steal Mode):** Figure 7-20 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and word-size full address mode transfer (cycle steal mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

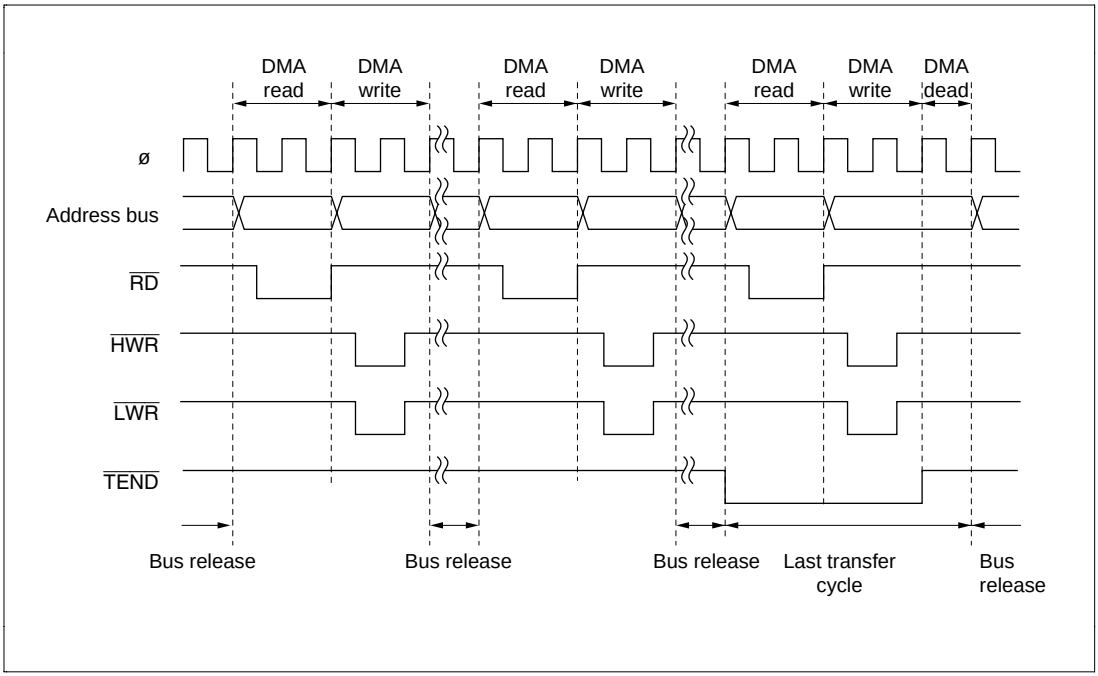



**Figure 7-20 Example of Full Address Mode (Cycle Steal) Transfer**

A one-byte or one-word transfer is performed, and after the transfer the bus is released. While the bus is released one bus cycle is inserted by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

**Full Address Mode (Burst Mode):** Figure 7-21 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and word-size full address mode transfer (burst mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

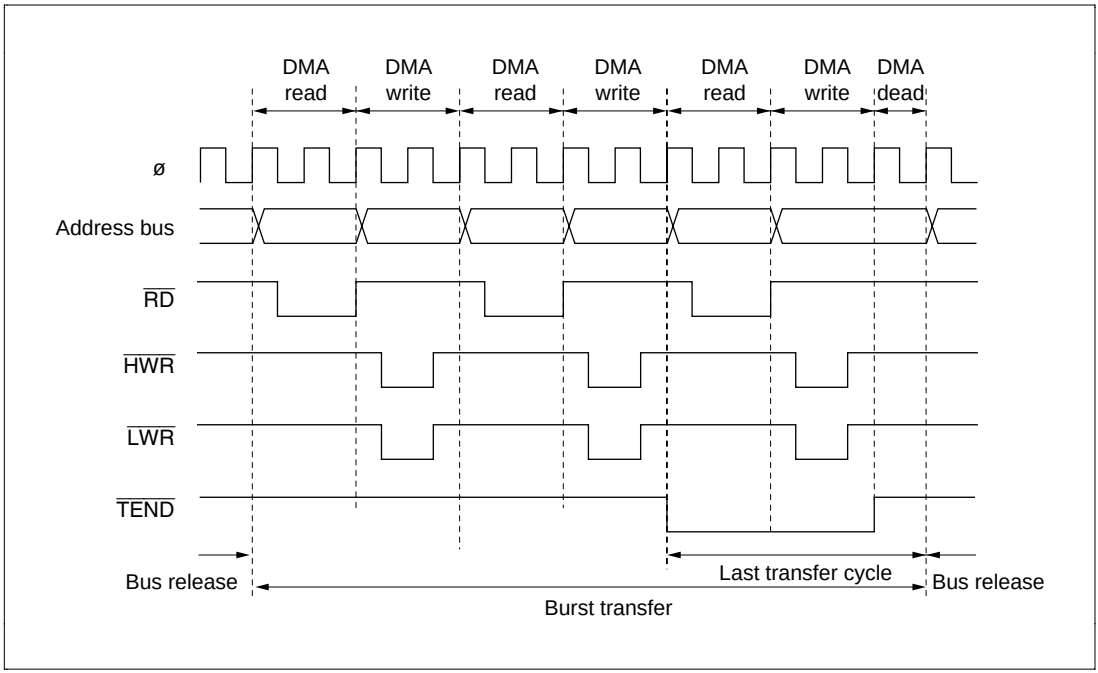



**Figure 7-21 Example of Full Address Mode (Burst Mode) Transfer**

In burst mode, one-byte or one-word transfers are executed consecutively until transfer ends.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

If a request from another higher-priority channel is generated after burst transfer starts, that channel has to wait until the burst transfer ends.

If an NMI is generated while a channel designated for burst transfer is in the transfer enabled state, the DTME bit is cleared and the channel is placed in the transfer disabled state. If burst transfer has already been activated inside the DMAC, the bus is released on completion of a one-byte or one-word transfer within the burst transfer, and burst transfer is suspended. If the last transfer cycle of the burst transfer has already been activated inside the DMAC, execution continues to the end of the transfer even if the DTME bit is cleared.

**Full Address Mode (Block Transfer Mode):** Figure 7-22 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and word-size full address mode transfer (block transfer mode) is performed from internal 16-bit, 1-state access space to external 16-bit, 2-state access space.

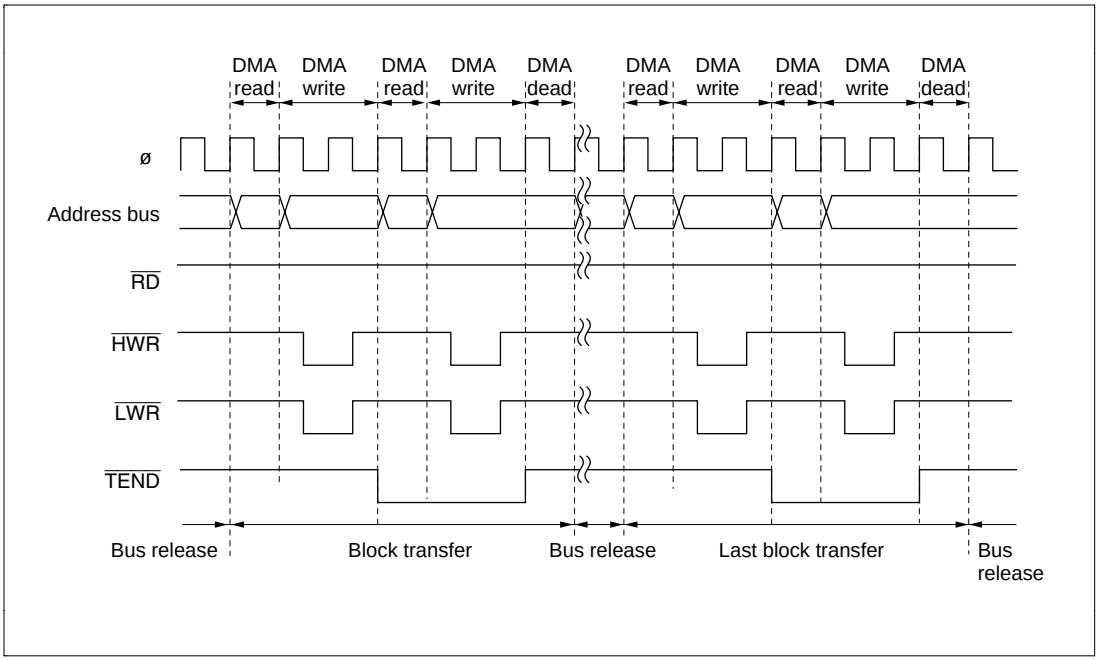



**Figure 7-22 Example of Full Address Mode (Block Transfer Mode) Transfer**

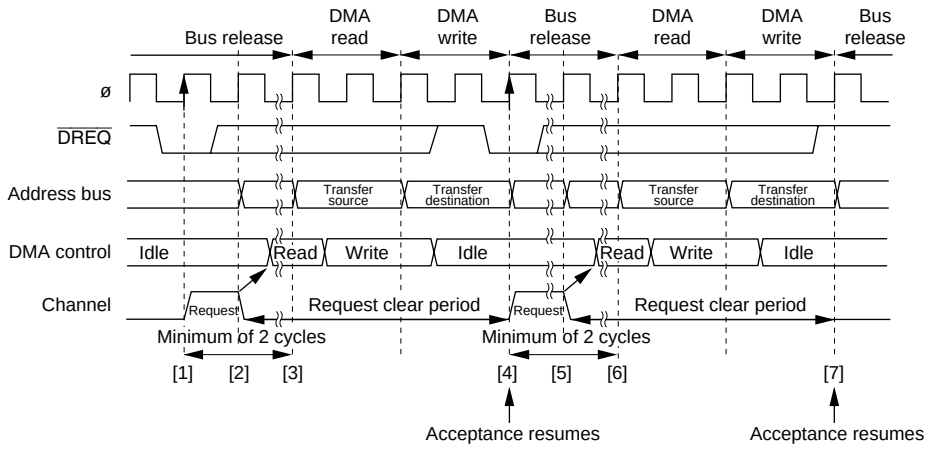
A one-block transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are inserted by the CPU or DTC.

In the transfer end cycle of each block (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

One block is transmitted without interruption. NMI generation does not affect block transfer operation.

**$\overline{\text{DREQ}}$ Pin Falling Edge Activation Timing:** Set the DTA bit for the channel for which the $\overline{\text{DREQ}}$ pin is selected to 1.

Figure 7-23 shows an example of $\overline{\text{DREQ}}$ pin falling edge activated normal mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.

[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.

[3] [6] Start of DMA cycle; $\overline{\text{DREQ}}$ pin high level sampling on the rising edge of ø starts.

[4] [7] When the $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after the write cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

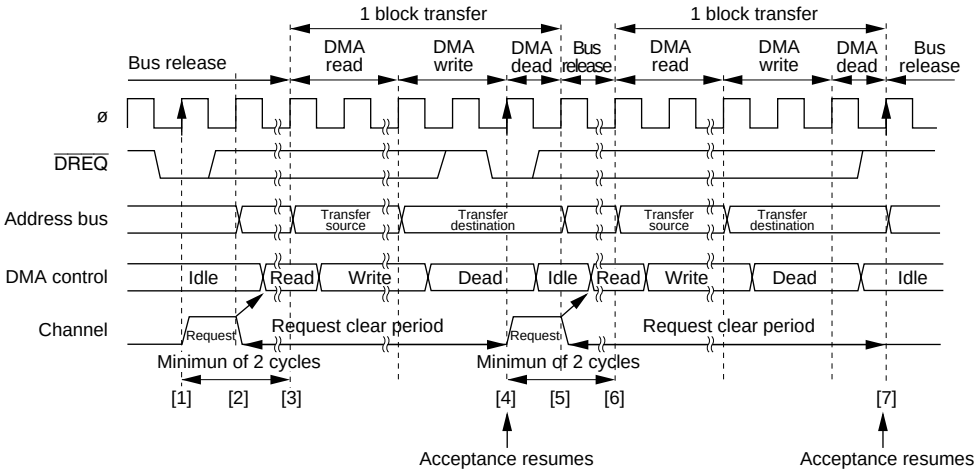
Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

**Figure 7-23 Example of $\overline{\text{DREQ}}$ Pin Falling Edge Activated Normal Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge detection is started. If $\overline{\text{DREQ}}$ pin high level sampling has been completed by the time the DMA write cycle ends, acceptance resumes after the end of the write cycle, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

Figure 7-24 shows an example of $\overline{\text{DREQ}}$ pin falling edge activated block transfer mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] Start of DMA cycle; $\overline{\text{DREQ}}$ pin high level sampling on the rising edge of ø starts.
[4] [7] When the $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after the dead cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

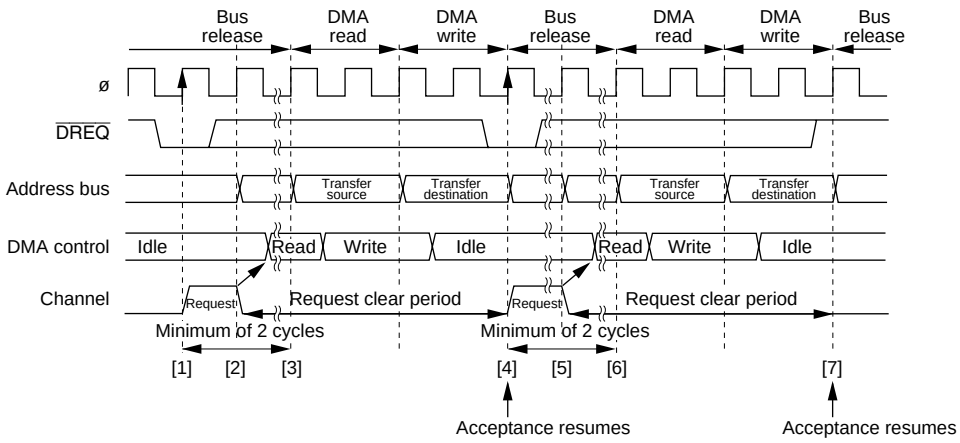
**Figure 7-24 Example of $\overline{\text{DREQ}}$ Pin Falling Edge Activated Block Transfer Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge detection is started. If $\overline{\text{DREQ}}$ pin high level sampling has been completed by the time the DMA dead cycle ends, acceptance resumes after the end of the dead cycle, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

**$\overline{\text{DREQ}}$ Level Activation Timing (Normal Mode):** Set the DTA bit for the channel for which the $\overline{\text{DREQ}}$ pin is selected to 1.

Figure 7-25 shows an example of $\overline{\text{DREQ}}$ level activated normal mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] The DMA cycle is started.
[4] [7] Acceptance is resumed after the write cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

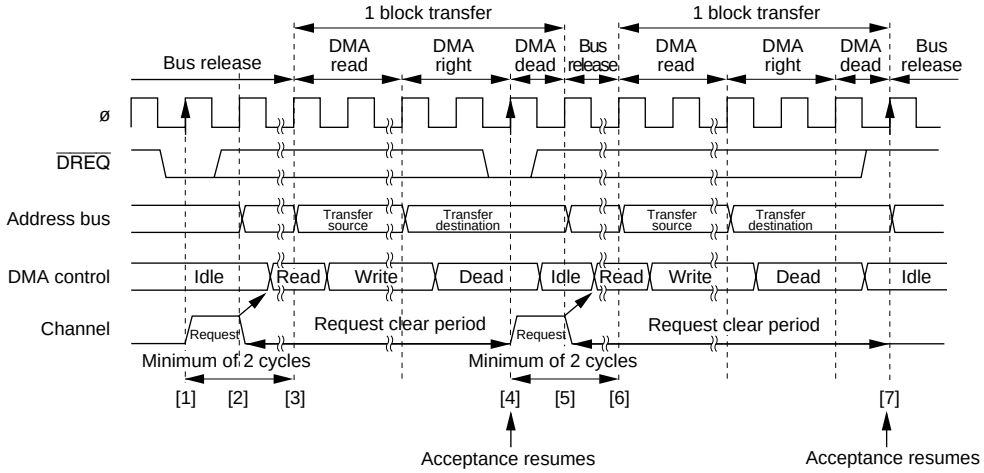
Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

**Figure 7-25 Example of $\overline{\text{DREQ}}$ Level Activated Normal Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the write cycle, acceptance resumes, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

Figure 7-26 shows an example of $\overline{\text{DREQ}}$ level activated block transfer mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] The DMA cycle is started.
[4] [7] Acceptance is resumed after the dead cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

**Figure 7-26 Example of $\overline{\text{DREQ}}$ Level Activated Block Transfer Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the dead cycle, acceptance resumes, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

### 7.5.11 DMAC Bus Cycles (Single Address Mode)

**Single Address Mode (Read):** Figure 7-27 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and byte-size single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.

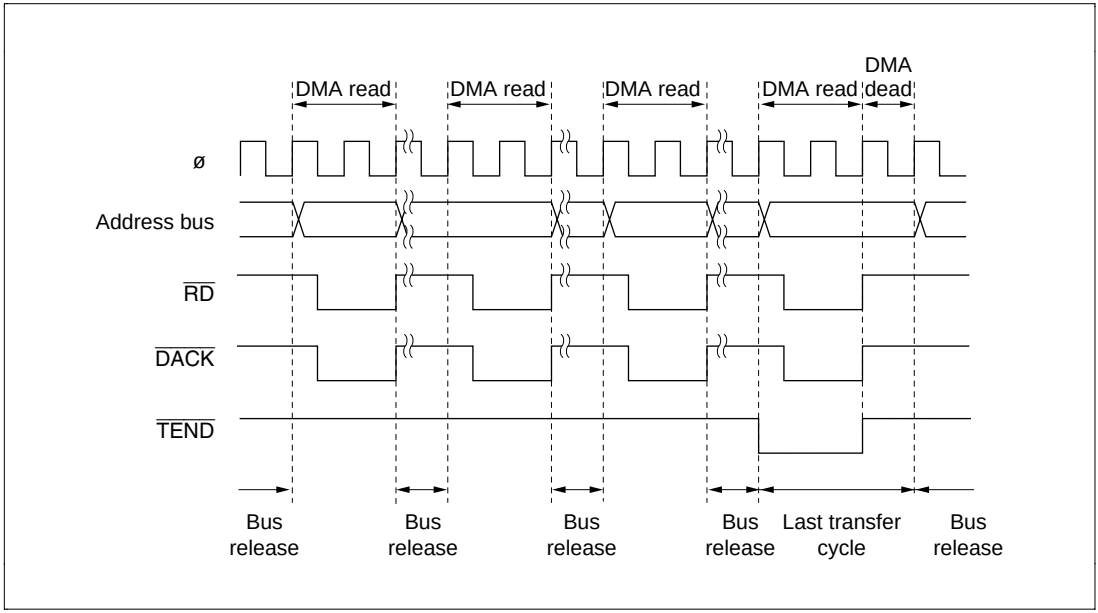



**Figure 7-27 Example of Single Address Mode (Byte Read) Transfer**

Figure 7-28 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and word-size single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.

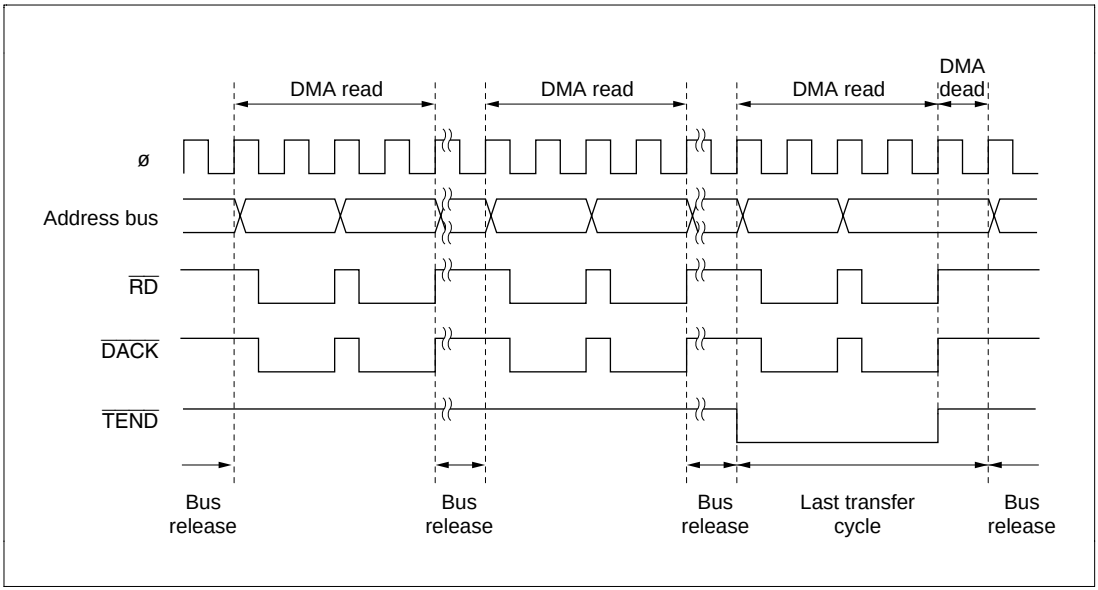



**Figure 7-28 Example of Single Address Mode (Word Read) Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are inserted by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

**Single Address Mode (Write):** Figure 7-29 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and byte-size single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.

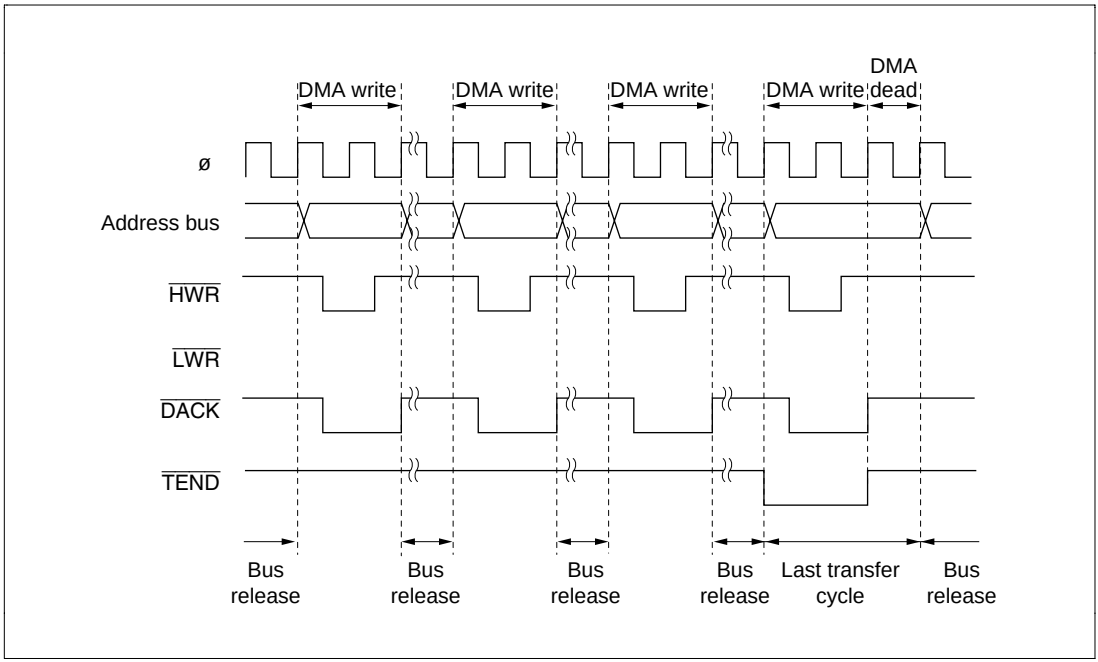



**Figure 7-29 Example of Single Address Mode (Byte Write) Transfer**

Figure 7-30 shows a transfer example in which $\overline{\text{TEND}}$ output is enabled and word-size single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.

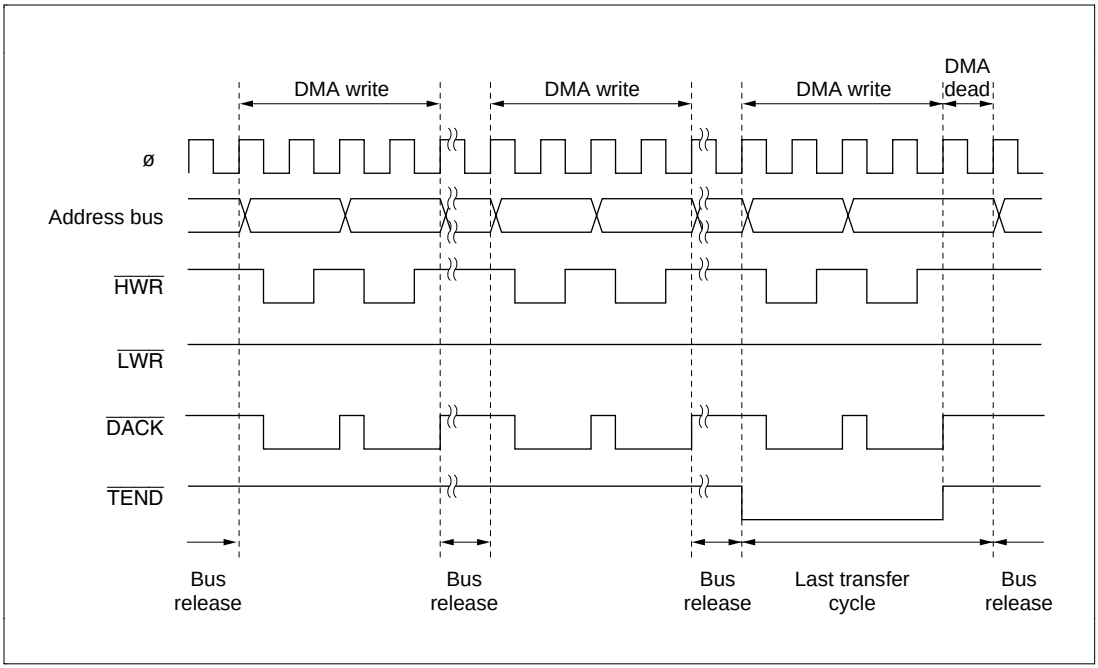


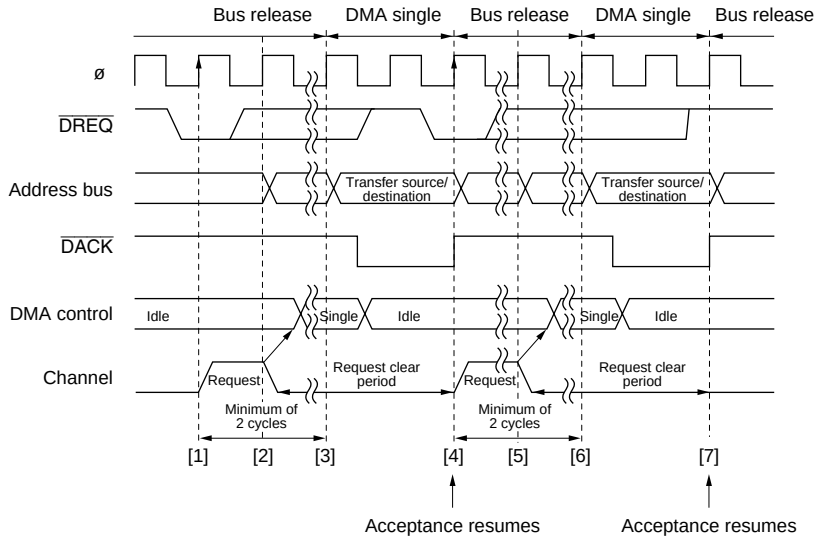

**Figure 7-30 Example of Single Address Mode (Word Write) Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released one or more bus cycles are inserted by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

**$\overline{\text{DREQ}}$ Pin Falling Edge Activation Timing:** Set the DTA bit for the channel for which the $\overline{\text{DREQ}}$ pin is selected to 1.

Figure 7-31 shows an example of $\overline{\text{DREQ}}$ pin falling edge activated single address mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] Start of DMA cycle; $\overline{\text{DREQ}}$ pin high level sampling on the rising edge of ø starts.
[4] [7] When the $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after the single cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

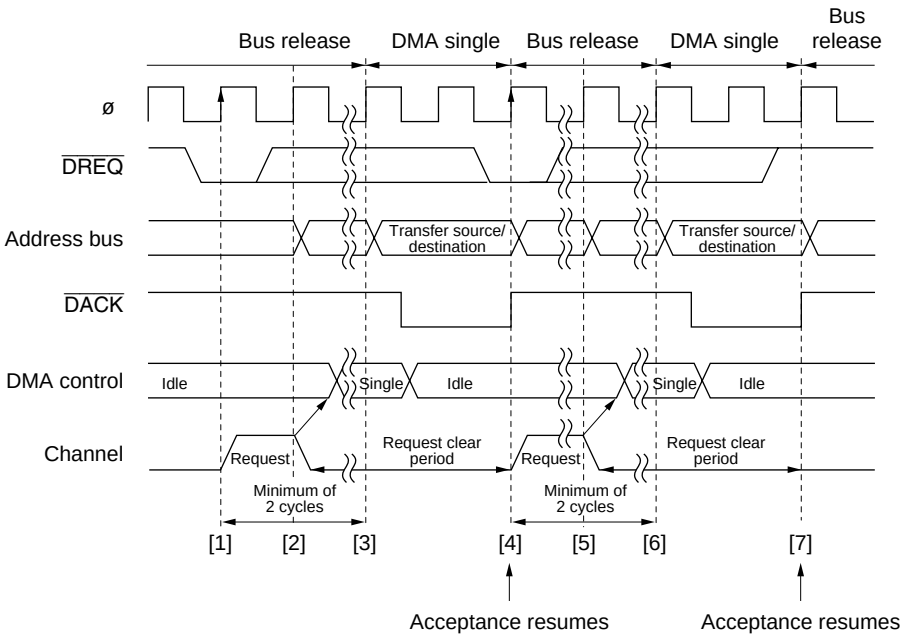
**Figure 7-31 Example of $\overline{\text{DREQ}}$ Pin Falling Edge Activated Single Address Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge detection is started. If $\overline{\text{DREQ}}$ pin high level sampling has been completed by the time the DMA single cycle ends, acceptance resumes after the end of the single cycle, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

**$\overline{\text{DREQ}}$ Pin Low Level Activation Timing:** Set the DTA bit for the channel for which the $\overline{\text{DREQ}}$ pin is selected to 1.

Figure 7-32 shows an example of $\overline{\text{DREQ}}$ pin low level activated single address mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] The DMAC cycle is started.
[4] [7] Acceptance is resumed after the single cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ pin low level is sampled on the rising edge of ø, and the request is held.)

Note: In write data buffer mode, bus breaks from [2] to [7] may be hidden, and not visible.

**Figure 7-32 Example of $\overline{\text{DREQ}}$ Pin Low Level Activated Single Address Mode Transfer**

$\overline{\text{DREQ}}$ pin sampling is performed every cycle, with the rising edge of the next ø cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ pin low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the single cycle, acceptance resumes, $\overline{\text{DREQ}}$ pin low level sampling is performed again, and this operation is repeated until the transfer ends.

### 7.5.12 Write Data Buffer Function

DMAC internal-to-external dual address transfers and single address transfers can be executed at high speed using the write data buffer function, enabling system throughput to be improved.

When the WDBE bit of BCRL in the bus controller is set to 1, enabling the write data buffer function, dual address transfer external write cycles or single address transfers and internal accesses (on-chip memory or internal I/O registers) are executed in parallel. Internal accesses are independent of the bus master, and DMAC dead cycles are regarded as internal accesses.

A low level can always be output from the $\overline{\text{TEND}}$ pin if the bus cycle in which a low level is to be output is an external bus cycle. However, a low level is not output from the $\overline{\text{TEND}}$ pin if the bus cycle in which a low level is to be output from the $\overline{\text{TEND}}$ pin is an internal bus cycle, and an external write cycle is executed in parallel with this cycle.

Figure 7-33 shows an example of burst mode transfer from on-chip RAM to external memory using the write data buffer function.

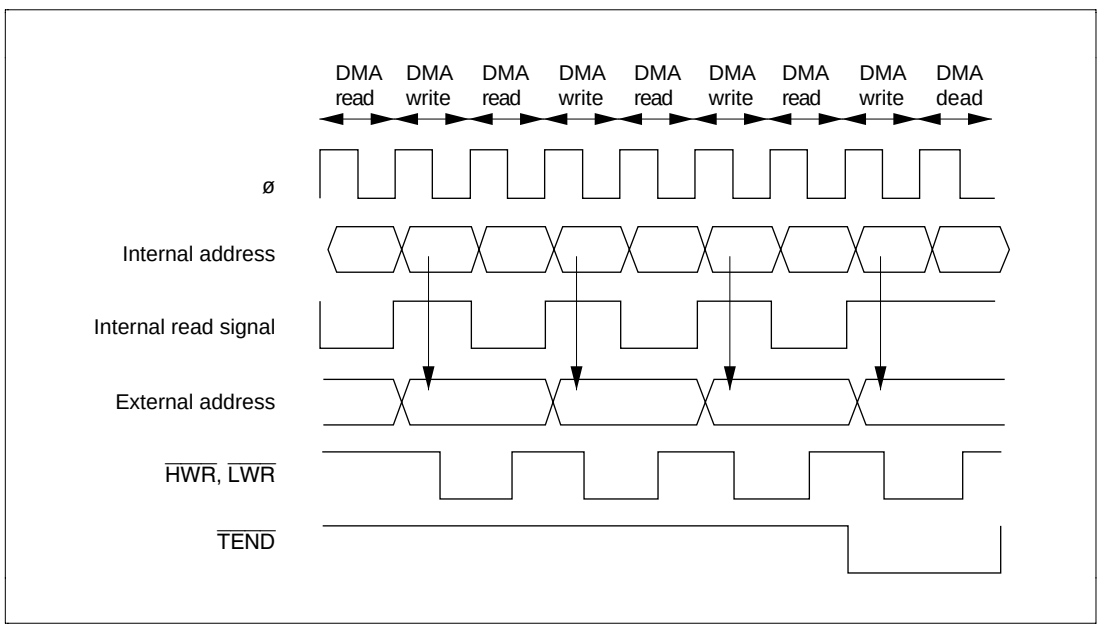



**Figure 7-33 Example of Dual Address Transfer Using Write Data Buffer Function**

Figure 7-34 shows an example of single address transfer using the write data buffer function. In this example, the CPU program area is in on-chip memory.

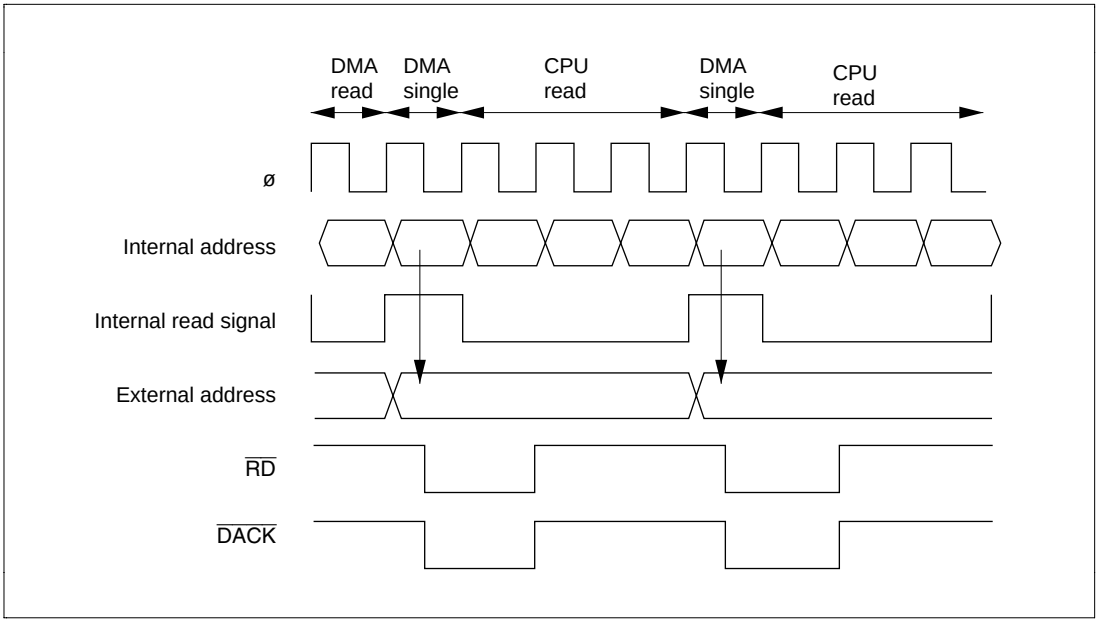



**Figure 7-34 Example of Single Address Transfer Using Write Data Buffer Function**

When the write data buffer function is activated, the DMAC recognizes that the bus cycle concerned has ended, and starts the next operation. Therefore, $\overline{\text{DREQ}}$ pin sampling is started one state after the start of the DMA write cycle or single address transfer.

### 7.5.13 DMAC Multi-Channel Operation

The DMAC channel priority order is: channel 0 > channel 1, and channel A > channel B. Table 7-13 summarizes the priority order for DMAC channels.

**Table 7-13 DMAC Channel Priority Order**

| Short Address Mode | Full Address Mode | Priority |
|---|---|---|
| Channel 0A | Channel 0 | High |
| Channel 0B | | |
| Channel 1A | Channel 1 | |
| Channel 1B | | Low |

If transfer requests are issued simultaneously for more than one channel, or if a transfer request for another channel is issued during a transfer, when the bus is released the DMAC selects the highest-priority channel from among those issuing a request according to the priority order shown in table 7-13.

During burst transfer, or when one block is being transferred in block transfer, the channel will not be changed until the end of the transfer.

Figure 7-35 shows a transfer example in which transfer requests are issued simultaneously for channels 0A, 0B, and 1.

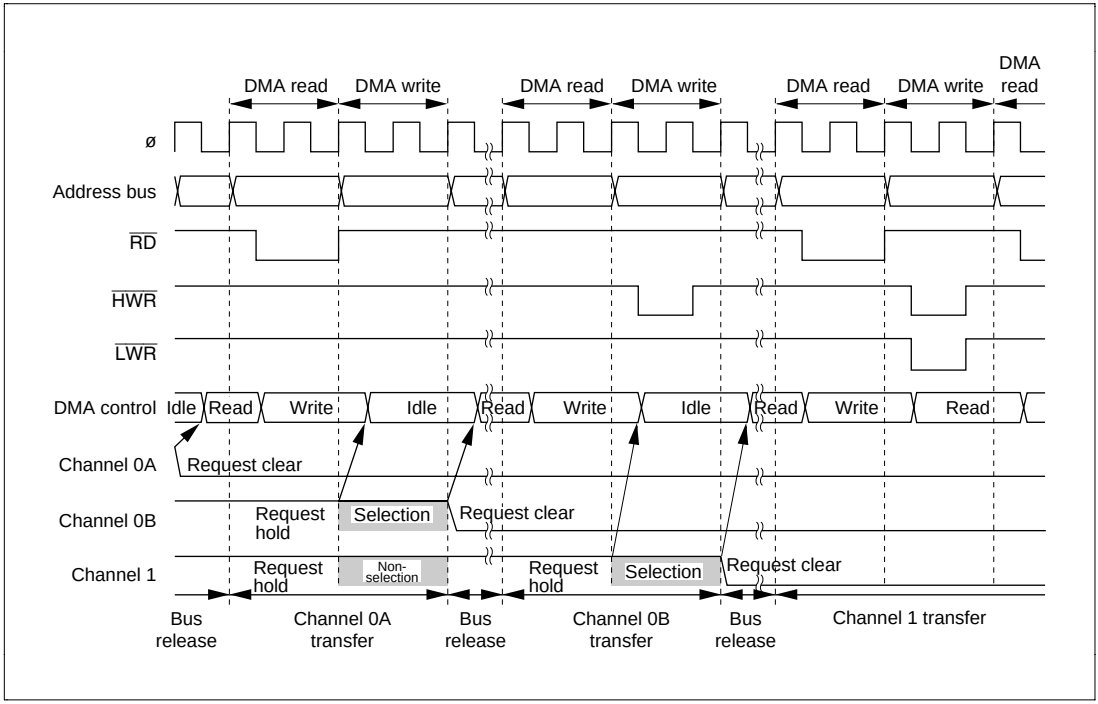



**Figure 7-35 Example of Multi-Channel Transfer**

### 7.5.14 Relation Between External Bus Requests, Refresh Cycles, the DTC, and the DMAC

There can be no break between a DMA cycle read and a DMA cycle write. This means that a refresh cycle, external bus release cycle, or DTC cycle is not generated between the external read and external write in a DMA cycle.

In the case of successive read and write cycles, such as in burst transfer or block transfer, a refresh or external bus released state may be inserted after a write cycle. Since the DTC has a lower priority than the DMAC, the DTC does not operate until the DMAC releases the bus.

When DMA cycle reads or writes are accesses to on-chip memory or internal I/O register, these DMA cycles may be executed at the same time as refresh cycles or external bus release.

### 7.5.15 NMI Interrupts and DMAC

When an NMI interrupt is requested, burst mode transfer in full address mode is interrupted. An NMI interrupt does not affect the operation of the DMAC in other modes.

In full address mode, transfer is enabled for a channel when both the DTE bit and the DTME bit are set to 1. With burst mode setting, the DTME bit is cleared when an NMI interrupt is requested.

If the DTME bit is cleared during burst mode transfer, the DMAC discontinues transfer on completion of the 1-byte or 1-word transfer in progress, then releases the bus, which passes to the CPU.

The channel on which transfer was interrupted can be restarted by setting the DTME bit to 1 again. Figure 7-36 shows the procedure for continuing transfer when it has been interrupted by an NMI interrupt on a channel designated for burst mode transfer.

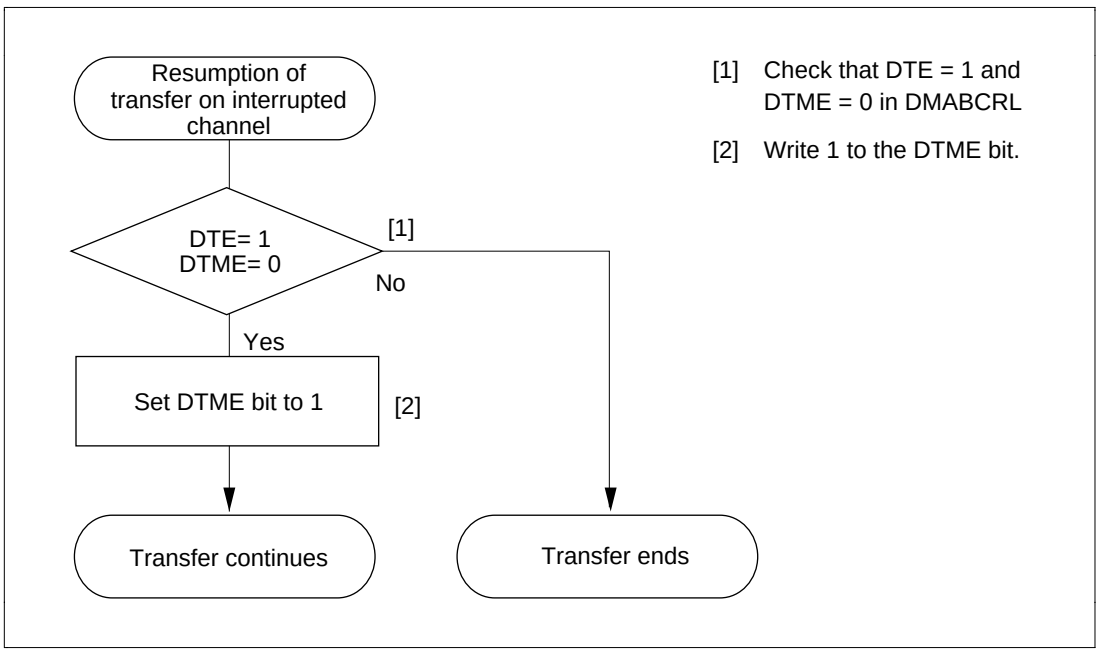



**Figure 7-36 Example of Procedure for Continuing Transfer on Channel Interrupted by NMI Interrupt**

### 7.5.16 Forced Termination of DMAC Operation

If the DTE bit for the channel currently operating is cleared to 0, the DMAC stops on completion of the 1-byte or 1-word transfer in progress. DMAC operation resumes when the DTE bit is set to 1 again.

In full address mode, the same applies to the DTME bit.

Figure 7-37 shows the procedure for forcibly terminating DMAC operation by software.

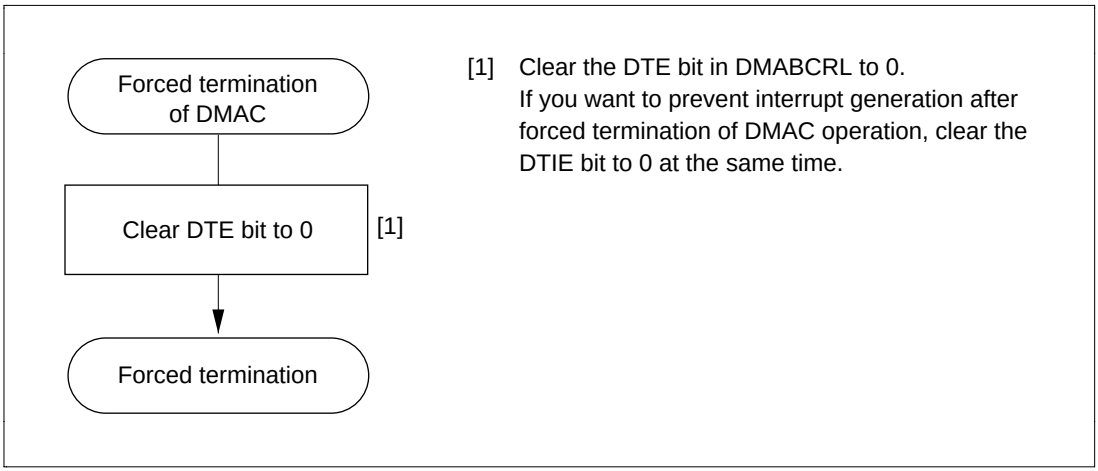



**Figure 7-37 Example of Procedure for Forcibly Terminating DMAC Operation**

### 7.5.17 Clearing Full Address Mode

Figure 7-38 shows the procedure for releasing and initializing a channel designated for full address mode. After full address mode has been cleared, the channel can be set to another transfer mode using the appropriate setting procedure.

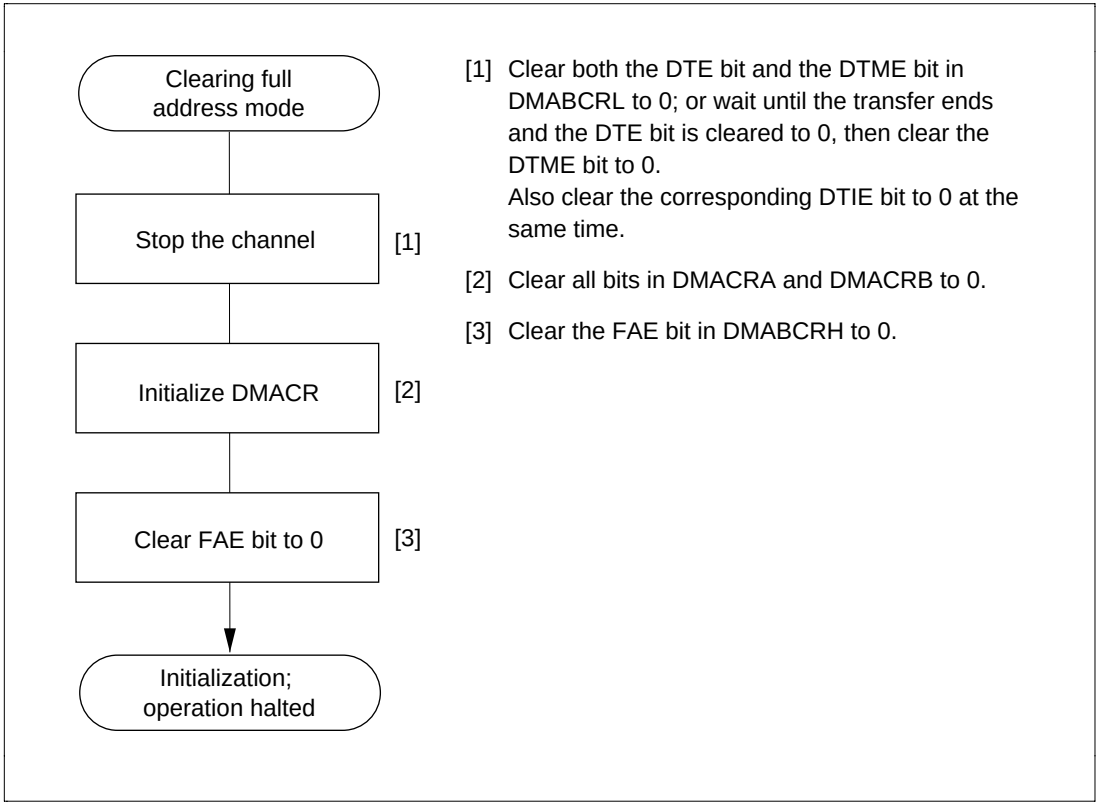



**Figure 7-38 Example of Procedure for Clearing Full Address Mode**

## 7.6 Interrupts

The sources of interrupts generated by the DMAC are transfer end and transfer break. Table 7-13 shows the interrupt sources and their priority order.

**Table 7-13 Interrupt Source Priority Order**

| Interrupt Name | Interrupt Source | | Interrupt Priority Order |
|---|---|---|---|
| | Short Address Mode | Full Address Mode | |
| DEND0A | Interrupt due to end of transfer on channel 0A | Interrupt due to end of transfer on channel 0 | High |
| DEND0B | Interrupt due to end of transfer on channel 0B | Interrupt due to break in transfer on channel 0 | |
| DEND1A | Interrupt due to end of transfer on channel 1A | Interrupt due to end of transfer on channel 1 | |
| DEND1B | Interrupt due to end of transfer on channel 1B | Interrupt due to break in transfer on channel 1 | Low |

Enabling or disabling of each interrupt source is set by means of the DTIE bit for the corresponding channel in DMABCR, and interrupts from each source are sent to the interrupt controller independently.

The relative priority of transfer end interrupts on each channel is decided by the interrupt controller, as shown in table 7-13.

Figure 7-39 shows a block diagram of a transfer end/transfer break interrupt. An interrupt is always generated when the DTIE bit is set to 1 while DTE bit is cleared to 0.

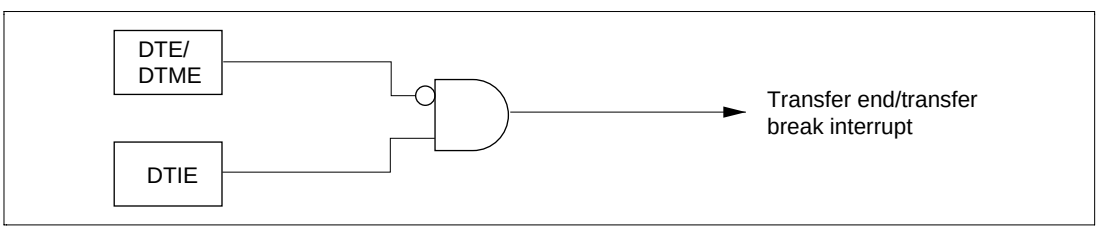



**Figure 7-39 Block Diagram of Transfer End/Transfer Break Interrupt**

In full address mode, a transfer break interrupt is generated when the DTME bit is cleared to o while DTIEB bit is set to 1.

In both short address mode and full address mode, DMABCR should be set so as to prevent the occurrence of a combination that constitutes a condition for interrupt generation during setting.

## 7.7 Usage Notes

**DMAC Register Access during Operation:** Except for forced termination, the operating (including transfer waiting state) channel setting should not be changed. The operating channel setting should only be changed when transfer is disabled.

Also, the DMAC register should not be written to in a DMA transfer.

**Module Stop:** When the MSTP15 bit in MSTPCR is set to 1, the DMAC clock stops, and the module stop state is entered. However, 1 cannot be written to the MSTP15 bit if any of the DMAC channels is enabled. This setting should therefore be made when DMAC operation is stopped.

When the DMAC clock stops, DMAC register accesses can no longer be made. Since the following DMAC register settings are valid even in the module stop state, they should be invalidated, if necessary, before a module stop.

- Transfer end/suspend interrupt (DTE = 0 and DTIE = 1)
- TEND pin enable (TEE = 1)
- $\overline{\text{DACK}}$ pin enable (FAE = 0 and SAE = 1)

**Medium-Speed Mode:** When the DTA bit is 0, internal interrupt signals specified as DMAC transfer sources are edge-detected.

In medium-speed mode, the DMAC operates on a medium-speed clock, while on-chip supporting modules operate on a high-speed clock. Consequently, if the period in which the relevant interrupt source is cleared by the CPU, DTC, or another DMAC channel, and the next interrupt is generated, is less than one state with respect to the DMAC clock (bus master clock), edge detection may not be possible and the interrupt may be ignored.

Also, in medium-speed mode, $\overline{\text{DREQ}}$ pin sampling is performed on the rising edge of the medium-speed clock.

**Write Data Buffer Function:** When the WDBE bit of BCRL in the bus controller is set to 1, enabling the write data buffer function, dual address transfer external write cycles or single address transfers and internal accesses (on-chip memory or internal I/O registers) are executed in parallel.

**(a) Write Data Buffer Function and DMAC Register Setting**

If the setting of  is changed during execution of an external access by means of the write data buffer function, the external access may not be performed normally.  The register that controls external accesses should only be manipulated when external reads, etc., are used with DMAC operation  disabled, and the operation is not performed in parallel with external access.\

**(b) Write Data Buffer Function and DMAC Operation Timing**

The DMAC can start its next operation during external access using the write data buffer function. Consequently, the $\overline{\text{DREQ}}$ pin sampling timing, $\overline{\text{TEND}}$ output timing, etc., are different from the case in which the write data buffer function is disabled.  Also, internal bus cycles maybe hidden, and not visible.

**(c) Write Data Buffer Function and $\overline{\text{TEND}}$ Output**

A low level is not output from the $\overline{\text{TEND}}$ pin if the bus cycle in which a low level is to be output from the $\overline{\text{TEND}}$ pin is an internal bus cycle, and an external write cycle is executed in parallel with this cycle.  Note, for example, that a low level may not be output from the $\overline{\text{TEND}}$ pin if the write data buffer function is used when data transfer is performed between an internal I/O register and on-chip memory.

If at least one of the DMAC transfer addresses is an external address, a low level is output from the $\overline{\text{TEND}}$ pin.

Figure 7-40 shows an example in which a low level is not output at the $\overline{\text{TEND}}$ pin.

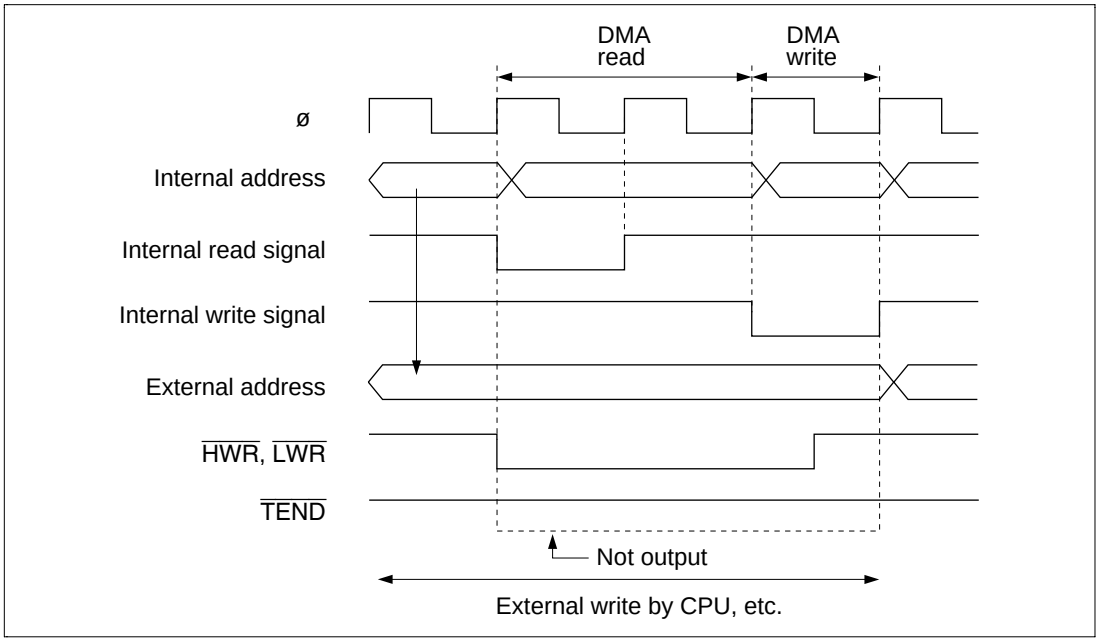



**Figure 7-40 Example in Which Low Level is Not Output at $\overline{\text{TEND}}$ Pin**

**Activation by Falling Edge on $\overline{\text{DREQ}}$ Pin:** $\overline{\text{DREQ}}$ pin falling edge detection is performed in synchronization with DMAC internal operations. The operation is as follows:

[1] Activation request wait state: Waits for detection of a low level on the $\overline{\text{DREQ}}$ pin, and switches to [2].

[2] Transfer wait state: Waits for DMAC data transfer to become possible, and switches to [3].

[3] Activation request disabled state: Waits for detection of a high level on the $\overline{\text{DREQ}}$ pin, and switches to [1].

After DMAC transfer is enabled, a transition is made to [1]. Thus, initial activation after transfer is enabled is performed by detection of a low level.

**Activation Source Acceptance:** At the start of activation source acceptance, a low level is detected in both $\overline{\text{DREQ}}$ pin falling edge sensing and low level sensing. Similarly, in the case of an internal interrupt, the interrupt request is detected. Therefore, a request is accepted from an internal interrupt or $\overline{\text{DREQ}}$ pin low level that occurs before execution of the DMABCRL write to enable transfer.

When the DMAC is activated, take any necessary steps to prevent an internal interrupt or $\overline{\text{DREQ}}$ pin low level remaining from the end of the previous transfer, etc.

**Internal Interrupt after End of Transfer:** When the DTE bit is cleared to 0 by the end of transfer or an abort, the selected internal interrupt request will be sent to the CPU or DTC even if DTA is set to 1.

Also, if internal DMAC activation has already been initiated when operation is aborted, the transfer is executed but flag clearing is not performed for the selected internal interrupt even if DTA is set to 1.

An internal interrupt request following the end of transfer or an abort should be handled by the CPU as necessary.

**Channel Re-Setting:** To reactivate a number of channels when multiple channels are enabled, use exclusive handling of transfer end interrupts, and perform DMABCR control bit operations exclusively.

Note, in particular, that in cases where multiple interrupts are generated between reading and writing of DMABCR, and a DMABCR operation is performed during new interrupt handling, the DMABCR write data in the original interrupt handling routine will be incorrect, and the write may invalidate the results of the operations by the multiple interrupts. Ensure that overlapping DMABCR operations are not performed by multiple interrupts, and that there is no separation between read and write operations by the use of a bit-manipulation instruction.

Also, when the DTE and DTME bits are cleared by the DMAC or are written with 0, they must first be read while cleared to 0 before the CPU can write a 1 to them.

# Section 8 Data Transfer Controller

## 8.1 Overview

The H8S/2655 Series includes a data transfer controller (DTC). The DTC can be activated by an interrupt or software, to transfer data.

### 8.1.1 Features

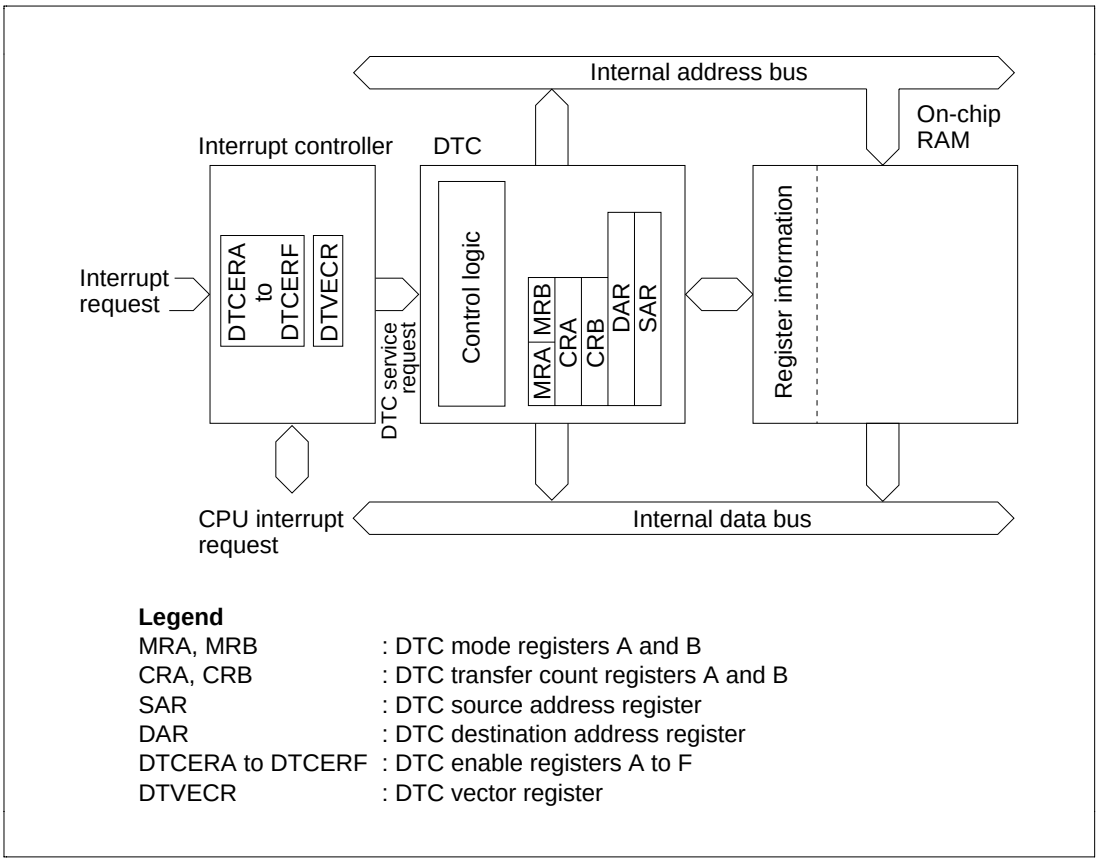
The features of the DTC are:

- Transfer possible over any number of channels
  - — Transfer information is stored in memory
  - — One activation source can trigger a number of data transfers (chain transfer)
- Wide range of transfer modes
  - — Normal, repeat, and block transfer modes available
  - — Incrementing, decrementing, and fixing of source and destination addresses can be selected
- Direct specification of 16-Mbyte address space possible
  - — 24-bit transfer source and destination addresses can be specified
- Transfer can be set in byte or word units
- A CPU interrupt can be requested for the interrupt that activated the DTC
  - — An interrupt request can be issued to the CPU after one data transfer ends
  - — An interrupt request can be issued to the CPU after the specified data transfers have completely ended
- Activation by software is possible

### 8.1.2 Block Diagram

Figure 8-1 shows a block diagram of the DTC.

The DTC's register information is stored in the on-chip RAM. A 32-bit bus connects the DTC to the on-chip RAM (1 kbyte), enabling 32-bit/1-state reading and writing of the DTC register information and hence helping to increase processing speed.




**Legend**

| | |
|---|---|
| MRA, MRB | : DTC mode registers A and B |
| CRA, CRB | : DTC transfer count registers A and B |
| SAR | : DTC source address register |
| DAR | : DTC destination address register |
| DTCERA to DTCERF | : DTC enable registers A to F |
| DTVECR | : DTC vector register |

**Figure 8-1 Block Diagram of DTC**

### 8.1.3 Register Configuration

Table 8-1 summarizes the DTC registers.

**Table 8-1 DTC Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| DTC mode register A | MRA | —*[2] | Undefined | —*[3] |
| DTC mode register B | MRB | —*[2] | Undefined | —*[3] |
| DTC source address register | SAR | —*[2] | Undefined | —*[3] |
| DTC destination address register | DAR | —*[2] | Undefined | —*[3] |
| DTC transfer count register A | CRA | —*[2] | Undefined | —*[3] |
| DTC transfer count register B | CRB | —*[2] | Undefined | —*[3] |
| DTC enable registers | DTCER | R/W | H'00 | H'FF30 to H'FF35 |
| DTC vector register | DTVECR | R/W | H'00 | H'FF37 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Registers within the DTC cannot be read or written to directly.
3. Addresses H'F800 to H'FBFF contain register information. When the DTC is used, do not clear the RAME bit in SYSCR to 0.

## 8.2 Register Descriptions

### 8.2.1 DTC Mode Register A (MRA)

MRA is an 8-bit register that controls the DTC operating mode.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz |
| Initial value | : | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | — | — | — |

**Bits 7 and 6—Source Address Mode 1 and 0 (SM1, SM0):** These bits specify whether SAR is to be incremented, decremented, or left fixed after a data transfer.

| Bit 7 | Bit 6 | |
|---|---|---|
| **SM1** | **SM0** | **Description** |
| 0 | — | SAR is fixed |
| 1 | 0 | SAR is incremented after a transfer<br>(by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | SAR is decremented after a transfer<br>(by –1 when Sz = 0; by –2 when Sz = 1) |

**Bits 5 and 4—Destination Address Mode 1 and 0 (DM1, DM0):** These bits specify whether DAR is to be incremented, decremented, or left fixed after a data transfer.

| Bit 5 | Bit 4 | |
|---|---|---|
| **DM1** | **DM0** | **Description** |
| 0 | — | DAR is fixed |
| 1 | 0 | DAR is incremented after a transfer<br>(by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | DAR is decremented after a transfer<br>(by –1 when Sz = 0; by –2 when Sz = 1) |

**Bits 3 and 2—DTC Mode (MD1, MD0):** These bits specify the DTC transfer mode.

| Bit 3 | Bit 2 | |
|---|---|---|
| **MD1** | **MD0** | **Description** |
| 0 | 0 | Normal mode |
| | 1 | Repeat mode |
| 1 | 0 | Block transfer mode |
| | 1 | — |

**Bit 1—DTC Transfer Mode Select (DTS):** Specifies whether the source side or the destination side is set to be a repeat area or block area, in repeat mode or block transfer mode.

| Bit 1 | |
|---|---|
| **DTS** | **Description** |
| 0 | Destination side is repeat area or block area |
| 1 | Source side is repeat area or block area |

**Bit 0—DTC Data Transfer Size (Sz):** Specifies the size of data to be transferred.

| Bit 0 | |
|---|---|
| **Sz** | **Description** |
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

### 8.2.2 DTC Mode Register B (MRB)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CHNE | DISEL | — | — | — | — | — | — |
| Initial value: | | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | — | — | — |

MRB is an 8-bit register that controls the DTC operating mode.

**Bit 7—DTC Chain Transfer Enable (CHNE):** Specifies chain transfer. With chain transfer, a number of data transfers can be performed consecutively in response to a single transfer request.

In data transfer with CHNE set to 1, determination of the end of the specified number of transfers, clearing of the interrupt source flag, and clearing of DTCER is not performed.

**Bit 7**

| CHNE | Description |
|---|---|
| 0 | End of DTC data transfer (activation waiting state is entered) |
| 1 | DTC chain transfer (new register information is read, then data is transferred) |

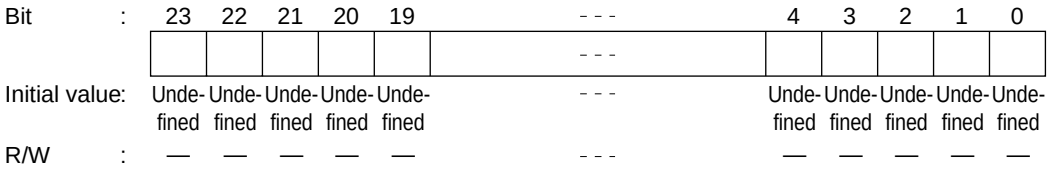
**Bit 6—DTC Interrupt Select (DISEL):** Specifies whether interrupt requests to the CPU are disabled or enabled after a data transfer.

**Bit 6**

| DISEL | Description |
|---|---|
| 0 | After a data transfer ends, the CPU interrupt is disabled unless the transfer counter is 0 (the DTC clears the interrupt source flag of the activating interrupt to 0) |
| 1 | After a data transfer ends, the CPU interrupt is enabled (the DTC does not clear the interrupt source flag of the activating interrupt to 0) |

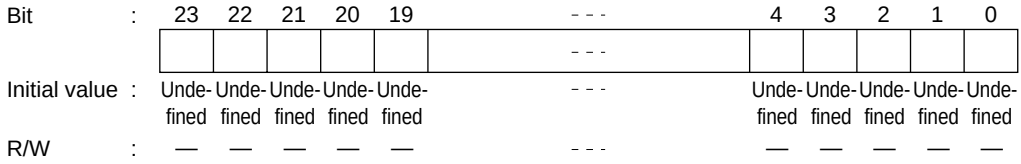
**Bits 5 to 0—Reserved:** These bits have no effect on DTC operation in the H8S/2655 Series, and should always be written with 0.

### 8.2.3 DTC Source Address Register (SAR)



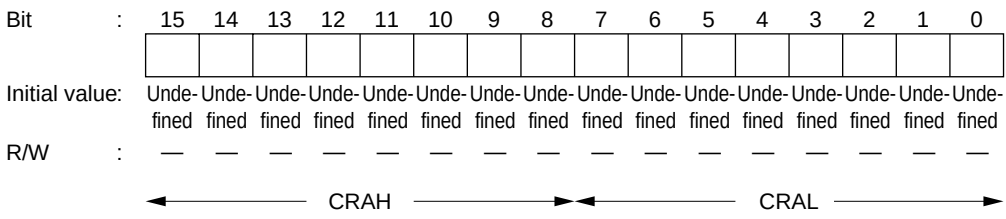

SAR is a 24-bit register that designates the source address of data to be transferred by the DTC. For word-size transfer, specify an even source address.

### 8.2.4 DTC Destination Address Register (DAR)




DAR is a 24-bit register that designates the destination address of data to be transferred by the DTC. For word-size transfer, specify an even destination address.

### 8.2.5 DTC Transfer Count Register A (CRA)




CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal mode, the entire CRA functions as a 16-bit transfer counter (1 to 65536). It is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

In repeat mode or block transfer mode, the CRA is divided into two parts: the upper 8 bits (CRAH) and the lower 8 bits (CRAL). CRAH holds the number of transfers while CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are sent when the count reaches H'00. This operation is repeated.

### 8.2.6 DTC Transfer Count Register B (CRB)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value: | | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W | : | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |

CRB is a 16-bit register that designates the number of times data is to be transferred by the DTC in block transfer mode. It functions as a 16-bit transfer counter (1 to 65536) that is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

### 8.2.7 DTC Enable Registers (DTCER)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The DTC enable registers comprise six 8-bit readable/writable registers, DTCERA to DTCERF, with bits corresponding to the interrupt sources that can activate the DTC. These bits enable or disable DTC service for the corresponding interrupt sources.

The DTC enable registers are initialized to H'00 by a reset and in hardware standby mode.

**Bit n—DTC Activation Enable (DTCEn)**

| Bit n<br>DTCEn | Description | |
|---|---|---|
| 0 | DTC activation by this interrupt is disabled<br>[Clearing conditions]<br>• When the DISEL bit is 1 and the data transfer has ended<br>• When the specified number of transfers have ended | (Initial value) |
| 1 | DTC activation by this interrupt is enabled<br>[Holding condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended | |

(n = 7 to 0)

A DTCE bit can be set for each interrupt source that can activate the DTC. The correspondence between interrupt sources and DTCE bits is shown in table 8-3, together with the vector number generated for each interrupt controller.

For DTCE bit setting, read/write operations must be performed using bit-manipulation instructions such as BSET and BCLR. For the initial setting only, however, when multiple activation sources are set at one time, it is possible to disable interrupts and write after executing a dummy read on the relevant register.

### 8.2.8 DTC Vector Register (DTVECR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SWDTE | DTVEC6 | DTVEC5 | DTVEC4 | DTVEC3 | DTVEC2 | DTVEC1 | DTVEC0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * A value of 1 can always be written to the SWDTE bit, but 0 can only be written after 1 is read.

DTVECR is an 8-bit readable/writable register that enables or disables DTC activation by software, and sets a vector number for the software activation interrupt.

DTVECR is initialized to H'00 by a reset and in hardware standby mode.

**Bit 7—DTC Software Activation Enable (SWDTE):** Enables or disables DTC activation by software.

When clearing the SWDTE bit to 0 by software, write 0 to SWDTE after reading SWDTE set to 1.

**Bit 7**

| SWDTE | Description |
|---|---|
| 0 | DTC software activation is disabled (Initial value)<br>[Clearing condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended |
| 1 | DTC software activation is enabled<br>[Holding conditions]<br>• When the DISEL bit is 1 and data transfer has ended<br>• When the specified number of transfers have ended<br>• During data transfer due to software activation |

**Bits 6 to 0—DTC Software Activation Vectors 6 to 0 (DTVEC6 to DTVEC0):** These bits specify a vector number for DTC software activation.

The vector address is expressed as H'0400 + ((vector number) << 1). <<1 indicates a one-bit left-shift. For example, when DTVEC6 to DTVEC0 = H'10, the vector address is H'0420.

### 8.2.9 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP14 bit in MSTPCR is set to 1, the DTC operation stops at the end of the bus cycle and a transition is made to module stop mode. However, 1 cannot be written in the MSTP14 bit while the DTC is operating. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 14—Module Stop (MSTP14):** Specifies the DTC module stop mode.

**Bit 14**

| MSTP14 | Description | |
|---|---|---|
| 0 | DTC module stop mode cleared | (Initial value) |
| 1 | DTC module stop mode set | |

## 8.3 Operation

### 8.3.1 Overview

When activated, the DTC reads register information that is already stored in memory and transfers data on the basis of that register information. After the data transfer, it writes updated register information back to memory. Pre-storage of register information in memory makes it possible to transfer data over any required number of channels. Setting the CHNE bit to 1 makes it possible to perform a number of transfers with a single activation.

Figure 8-2 shows a flowchart of DTC operation.

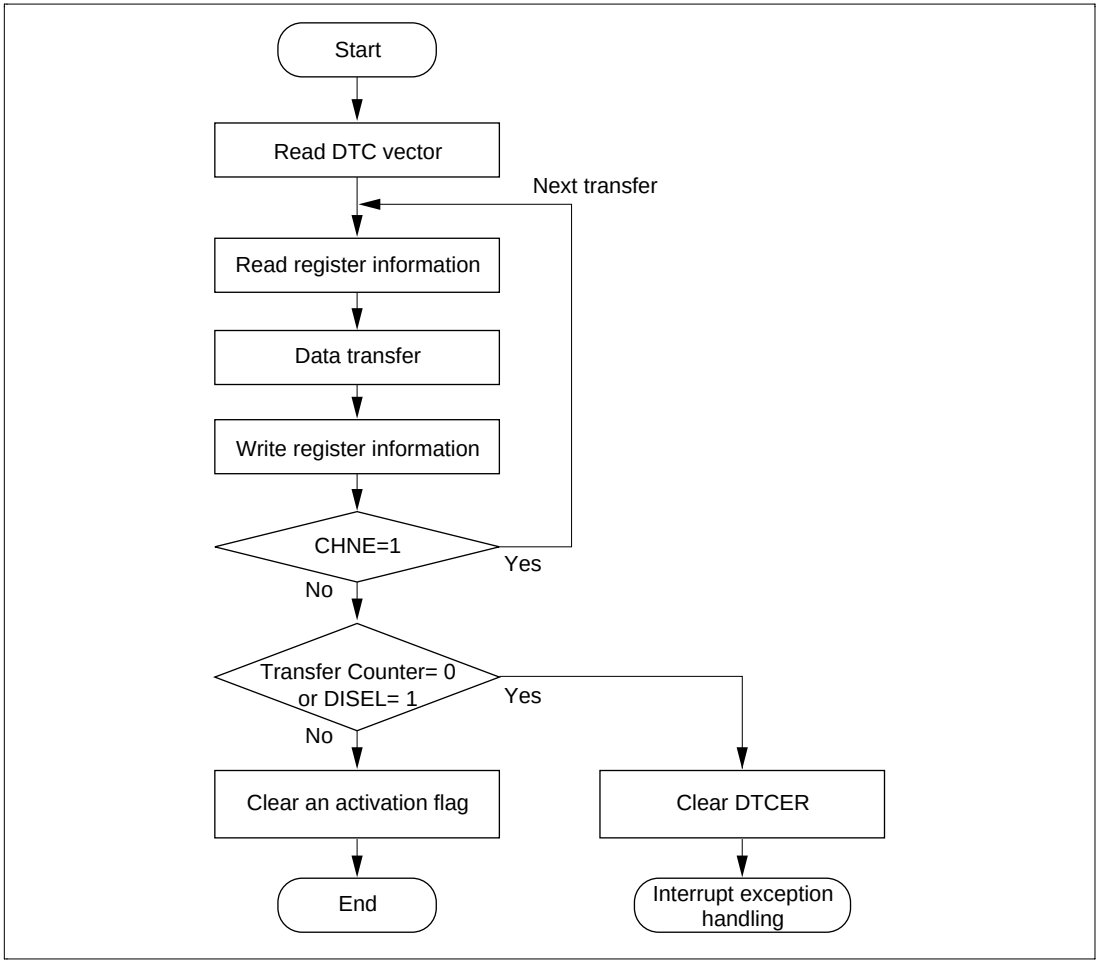



**Figure 8-2 Flowchart of DTC Operation**

The DTC transfer mode can be normal mode, repeat mode, or block transfer mode.

The 24-bit SAR designates the DTC transfer source address and the 24-bit DAR designates the transfer destination address. After each transfer, SAR and DAR are independently incremented, decremented, or left fixed.

Table 8-2 outlines the functions of the DTC.

**Table 8-2 DTC Functions**

| | | Address Registers | |
|---|---|---|---|
| **Transfer Mode** | **Activation Source** | **Transfer Source** | **Transfer Destination** |
| • Normal mode<br>— One transfer request transfers one byte or one word<br>— Memory addresses are incremented or decremented by 1 or 2<br>— Up to 65,536 transfers possible<br>• Repeat mode<br>— One transfer request transfers one byte or one word<br>— Memory addresses are incremented or decremented by 1 or 2<br>— After the specified number of transfers (1 to 256), the initial state resumes and operation continues<br>• Block transfer mode<br>— One transfer request transfers a block of the specified size<br>— Block size is from 1 to 256 bytes or words<br>— Up to 65,536 transfers possible<br>— A block area can be designated at either the source or destination | • IRQ<br>• TPU TGI<br>• 8-bit timer CMI<br>• SCI TXI or RXI<br>• A/D converter ADI<br>• DMAC DEND<br>• Software | 24 bits | 24 bits |

### 8.3.2 Activation Sources

The DTC operates when activated by an interrupt or by a write to DTVECR by software. An interrupt request can be directed to the CPU or DTC, as designated by the corresponding DTCER bit. An interrupt becomes a DTC activation source when the corresponding bit is set to 1, and a CPU interrupt source when the bit is cleared to 0.

At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source or corresponding DTCER bit is cleared. Table 8-3 shows activation source and DTCER clearance. The activation source flag, in the case of RXI0, for example, is the RDRF flag of SCI0.

**Table 8-3 Activation Source and DTCER Clearance**

| Activation Source | When the DISEL Bit Is 0 and the Specified Number of Transfers Have Not Ended | When the DISEL Bit Is 1, or when the Specified Number of Transfers Have Ended |
|---|---|---|
| Software activation | The SWDTE bit is cleared to 0 | The SWDTE bit remains set to 1<br>An interrupt is issued to the CPU |
| Interrupt activation | The corresponding DTCER bit remains set to 1<br>The activation source flag is cleared to 0 | The corresponding DTCER bit is cleared to 0<br>The activation source flag remains set to 1<br>A request is issued to the CPU for the activation source interrupt |

Figure 8-3 shows a block diagram of activation source control. For details see section 5, Interrupt Controller.

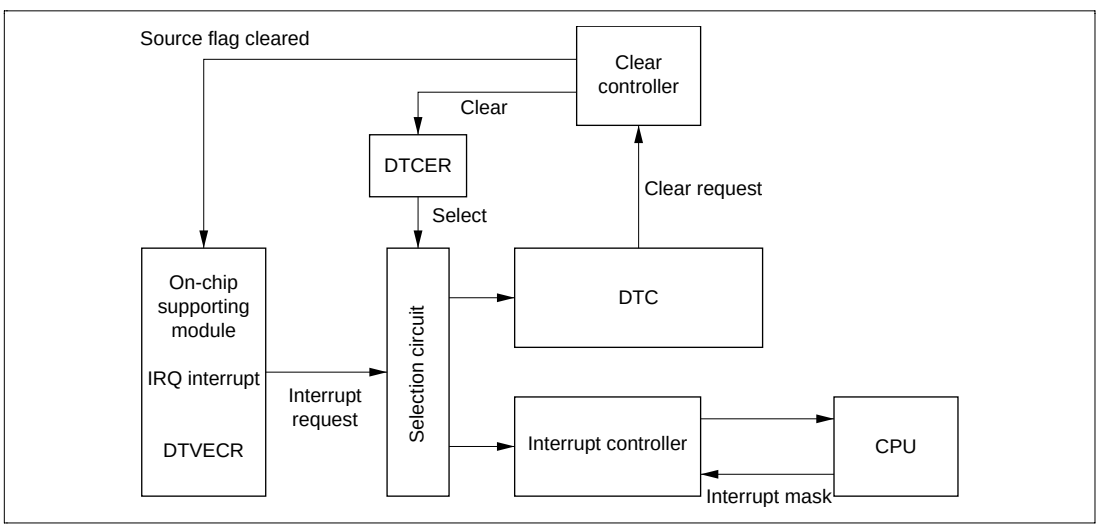



**Figure 8-3 Block Diagram of DTC Activation Source Control**

When an interrupt has been designated a DTC activation source, existing CPU mask level and interrupt controller priorities have no effect. If there is more than one activation source at the same time, the DTC operates in accordance with the default priorities.

### 8.3.3 DTC Vector Table

Figure 8-4 shows the correspondence between DTC vector addresses and register information.

Table 8-4 shows the correspondence between activation, vector addresses, and DTCER bits. When the DTC is activated by software, the vector address is obtained from: H'0400 + DTVECR[6:0] << 1 (where << 1 indicates a 1-bit left shift). For example, if DTVECR is H'10, the vector address is H'0420.

The DTC reads the start address of the register information from the vector address set for each activation source, and then reads the register information from that start address. The register information can be placed at predetermined addresses in the on-chip RAM. The start address of the register information should be an integral multiple of four.

The configuration of the vector address is the same in both normal and advanced modes, a 2-byte unit being used in both cases. These two bytes specify the lower bits of the address in the on-chip RAM.

**Table 8-4 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address | DTCE | Priority |
|---|---|---|---|---|---|
| Write to DTVECR | Software | DTVECR | H'0400+ (DTVECR [6:0] <<1) | — | High |
| IRQ0 | External pin | 16 | H'0420 | DTCEA7 | |
| IRQ1 | | 17 | H'0422 | DTCEA6 | |
| IRQ2 | | 18 | H'0424 | DTCEA5 | |
| IRQ3 | | 19 | H'0426 | DTCEA4 | |
| IRQ4 | | 20 | H'0428 | DTCEA3 | |
| IRQ5 | | 21 | H'042A | DTCEA2 | |
| IRQ6 | | 22 | H'042C | DTCEA1 | |
| IRQ7 | | 23 | H'042E | DTCEA0 | |
| ADI (A/D conversion end) | A/D | 28 | H'0438 | DTCEB6 | |
| TGI0A (GR0A compare match/ input capture) | TPU channel 0 | 32 | H'0440 | DTCEB5 | |
| TGI0B (GR0B compare match/ input capture) | | 33 | H'0442 | DTCEB4 | |
| TGI0C (GR0C compare match/ input capture) | | 34 | H'0444 | DTCEB3 | |
| TGI0D (GR0D compare match/ input capture) | | 35 | H'0446 | DTCEB2 | |
| TGI1A (GR1A compare match/ input capture) | TPU channel 1 | 40 | H'0450 | DTCEB1 | |
| TGI1B (GR1B compare match/ input capture) | | 41 | H'0452 | DTCEB0 | |
| TGI2A (GR2A compare match/ input capture) | TPU channel 2 | 44 | H'0458 | DTCEC7 | |
| TGI2B (GR2B compare match/ input capture) | | 45 | H'045A | DTCEC6 | Low |

**Table 8-4 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address | DTCE | Priority |
|---|---|---|---|---|---|
| TGI3A (GR3A compare match/ input capture) | TPU channel 3 | 48 | H'0460 | DTCEC5 | High |
| TGI3B (GR3B compare match/ input capture) | | 49 | H'0462 | DTCEC4 | |
| TGI3C (GR3C compare match/ input capture) | | 50 | H'0464 | DTCEC3 | |
| TGI3D (GR3D compare match/ input capture) | | 51 | H'0466 | DTCEC2 | |
| TGI4A (GR4A compare match/ input capture) | TPU channel 4 | 56 | H'0470 | DTCEC1 | |
| TGI4B (GR4B compare match/ input capture) | | 57 | H'0472 | DTCEC0 | |
| TGI5A (GR5A compare match/ input capture) | TPU channel 5 | 60 | H'0478 | DTCED5 | |
| TGI5B (GR5B compare match/ input capture) | | 61 | H'047A | DTCED4 | |
| CMIA0 | 8-bit timer channel 0 | 64 | H'0480 | DTCED3 | |
| CMIB0 | | 65 | H'0482 | DTCED2 | |
| CMIA1 | 8-bit timer channel 1 | 68 | H'0488 | DTCED1 | |
| CMIB1 | | 69 | H'048A | DTCED0 | |
| DMTEND0A (DMAC transfer end 0) | DMAC | 72 | H'0490 | DTCEE7 | |
| DMTEND0B (DMAC transfer end 1) | | 73 | H'0492 | DTCEE6 | |
| DMTEND1A (DMAC transfer end 2) | | 74 | H'0494 | DTCEE5 | |
| DMTEND1B (DMAC transfer end 3) | | 75 | H'0496 | DTCEE4 | |
| RXI0 (reception complete 0) | SCI channel 0 | 81 | H'04A2 | DTCEE3 | |
| TXI0 (transmit data empty 0) | | 82 | H'04A4 | DTCEE2 | |
| RXI1 (reception complete 1) | SCI channel 1 | 85 | H'04AA | DTCEE1 | |
| TXI1 (transmit data empty 1) | | 86 | H'04AC | DTCEE0 | |
| RXI2 (reception complete 2) | SCI channel 2 | 89 | H'04B2 | DTCEF7 | |
| TXI2 (transmit data empty 2) | | 90 | H'04B4 | DTCEF6 | Low |

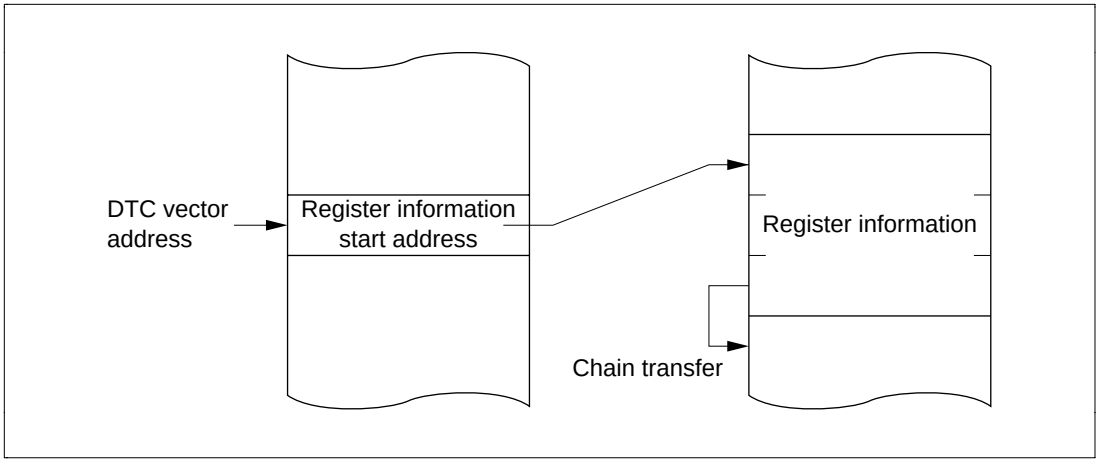



**Figure 8-4 Correspondence between DTC Vector Address and Register Information**

### 8.3.4 Location of Register Information in Address Space

Figure 8-5 shows how the register information should be located in the address space.

Locate the MRA, SAR, MRB, DAR, CRA, and CRB registers, in that order, from the start address of the register information (contents of the vector address). In the case of chain transfer, register information should be located in consecutive areas.

Locate the register information in the on-chip RAM (addresses: H'FFF800 to H'FFFBFF).

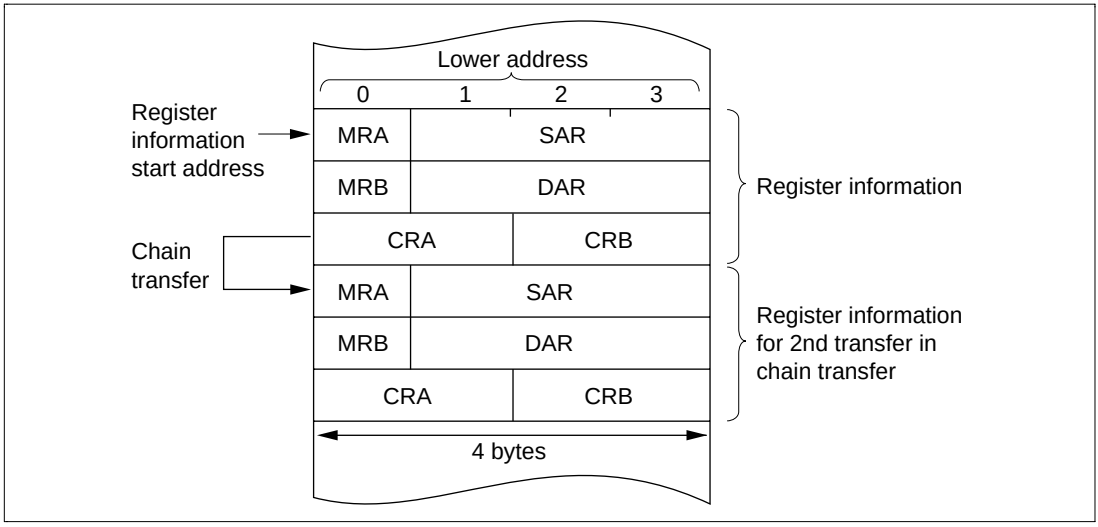



**Figure 8-5 Location of Register Information in Address Space**

### 8.3.5 Normal Mode

In normal mode, one operation transfers one byte or one word of data.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt can be requested.

Table 8-5 lists the register information in normal mode and figure 8-6 shows memory mapping in normal mode.

**Table 8-5 Register Information in Normal Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register A | CRA | Designates transfer count |
| DTC transfer count register B | CRB | Not used |

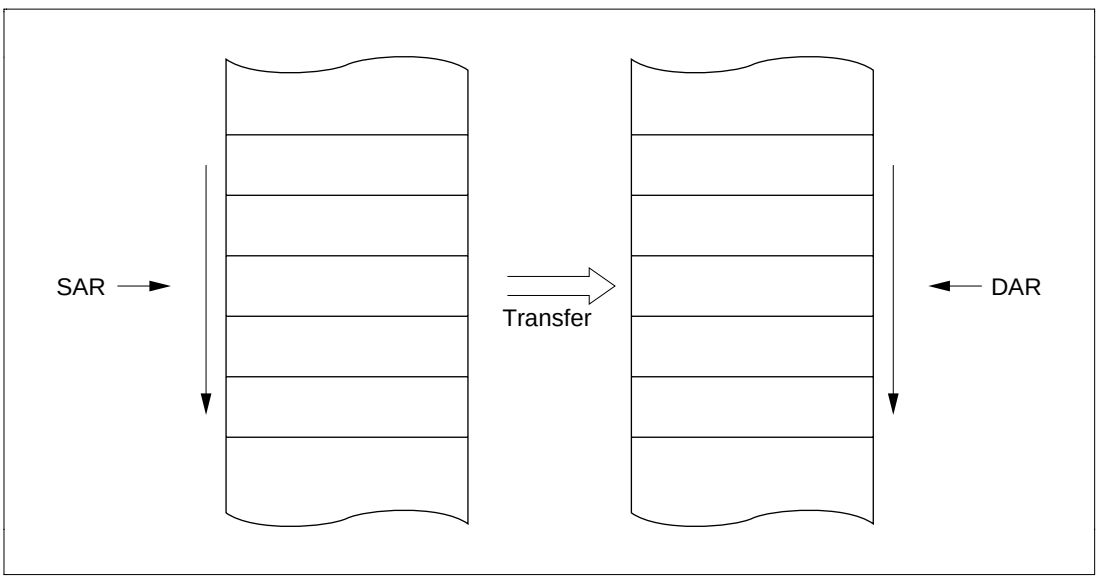



**Figure 8-6 Memory Mapping in Normal Mode**

### 8.3.6 Repeat Mode

In repeat mode, one operation transfers one byte or one word of data.

From 1 to 256 transfers can be specified. Once the specified number of transfers have ended, the initial state of the transfer counter and the address register specified as the repeat area is restored. In repeat mode the transfer counter value does not reach H'00, and therefore CPU interrupts cannot be requested when DISEL = 0.

Table 8-6 lists the register information in repeat mode and figure 8-7 shows memory mapping in repeat mode.

**Table 8-6 Register Information in Repeat Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register AH | CRAH | Holds number of transfers |
| DTC transfer count register AL | CRAL | Designates transfer count (8 bits × 2) |
| DTC transfer count register B | CRB | Not used |

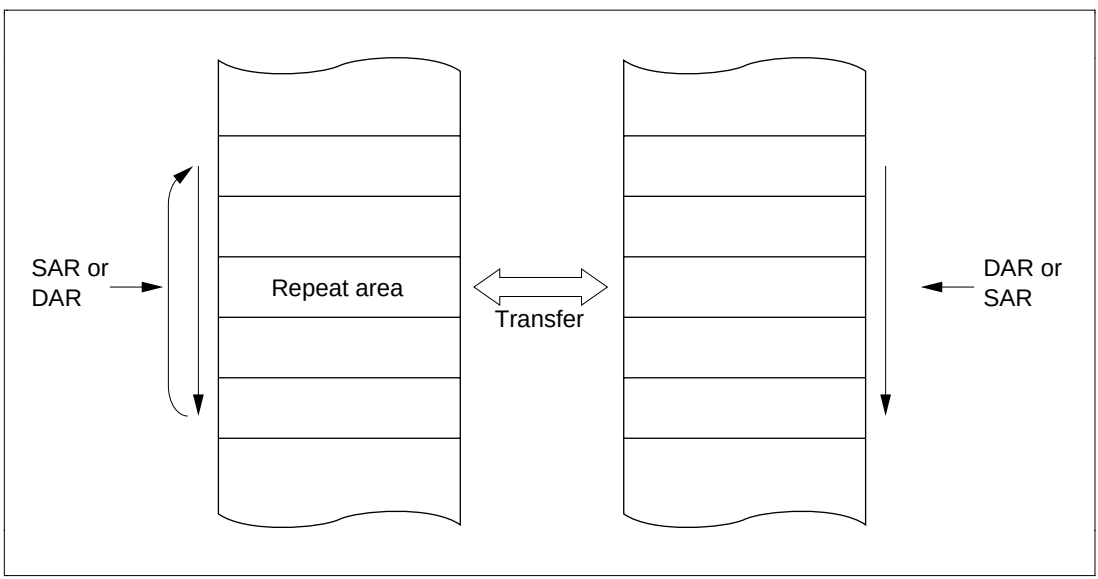



**Figure 8-7 Memory Mapping in Repeat Mode**

### 8.3.7 Block Transfer Mode

In block transfer mode, one operation transfers one block of data.

The block size is 1 to 256. When the transfer of one block ends, the initial state of the block size counter and the address register specified as the repeat area is restored. The other address register is then incremented, decremented, or left fixed.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt is requested.

Table 8-7 lists the register information in block transfer mode and figure 8-8 shows memory mapping in block transfer mode.

**Table 8-7 Register Information in Block Transfer Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates transfer source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register AH | CRAH | Holds block size |
| DTC transfer count register AL | CRAL | Designates block size count |
| DTC transfer count register B | CRB | Transfer count |

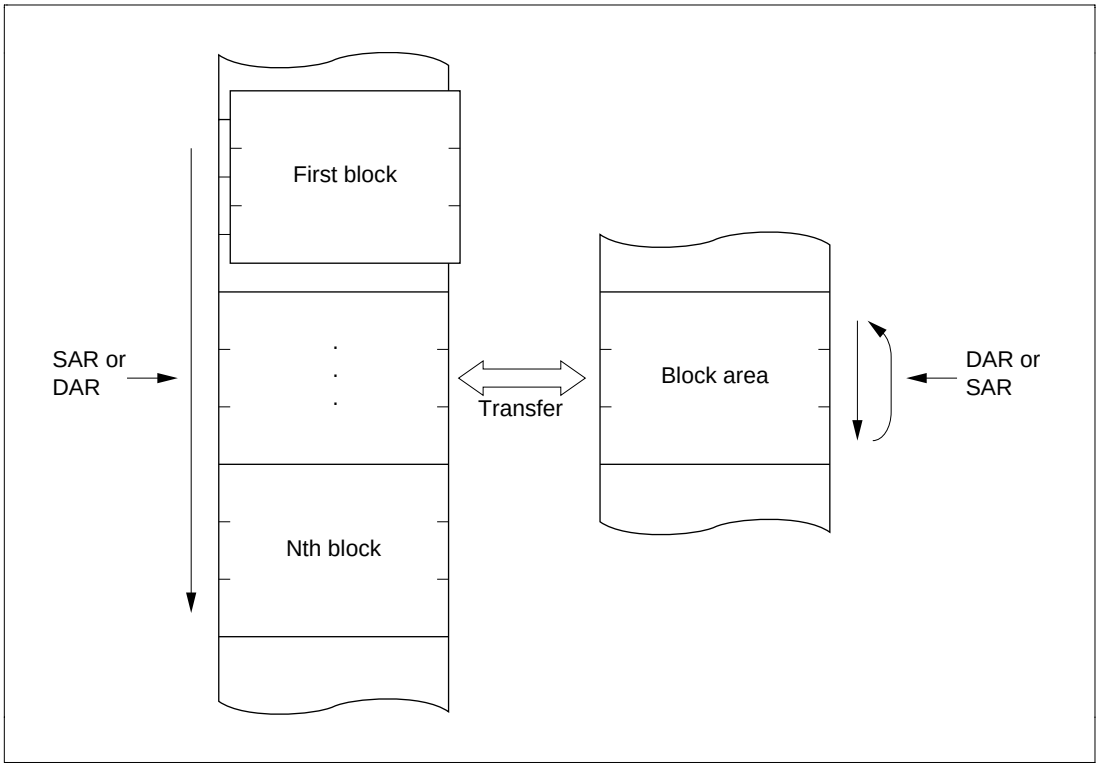



**Figure 8-8 Memory Mapping in Block Transfer Mode**

### 8.3.8 Chain Transfer

Setting the CHNE bit to 1 enables a number of data transfers to be performed consectutively in response to a single transfer request. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently.

Figure 8-9 shows the memory map for chain transfer.

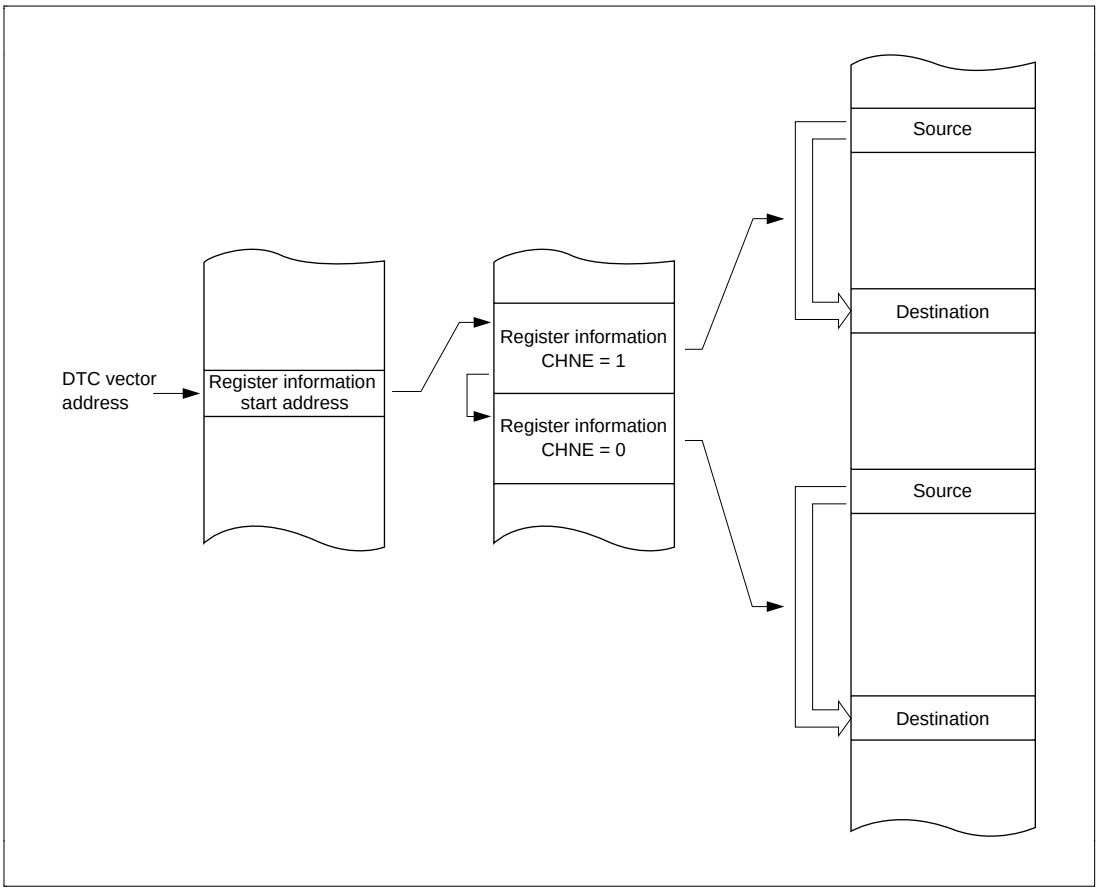



**Figure 8-9 Chain Transfer Memory Map**

In the case of transfer with CHNE set to 1, an interrupt request to the CPU is not generated at the end of the specified number of transfers or by setting of the DISEL bit to 1, and the interrupt source flag for the activation source is not affected.

### 8.3.9 Operation Timing

Figures 8-10 to 8-12 show an example of DTC operation timing.

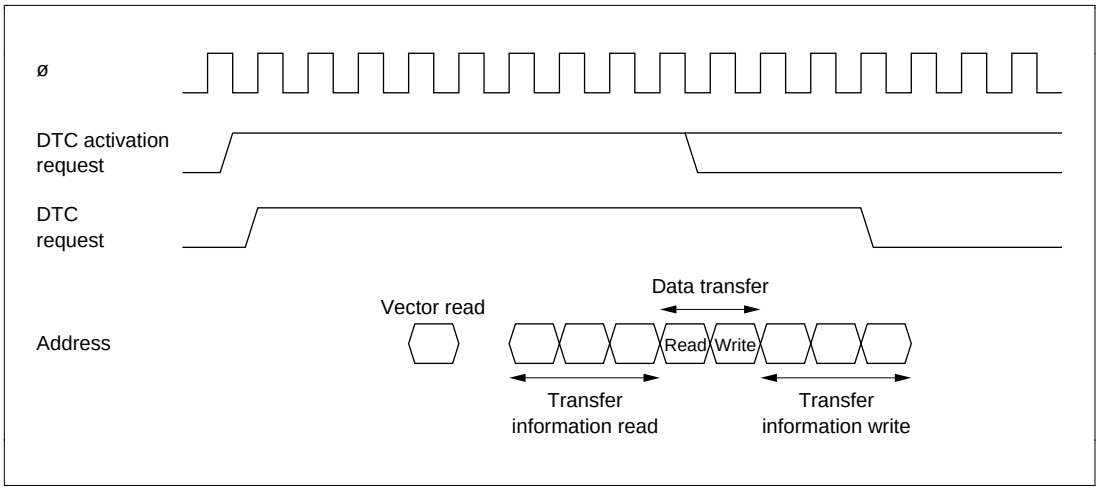



**Figure 8-10 DTC Operation Timing (Example in Normal Mode or Repeat Mode)**

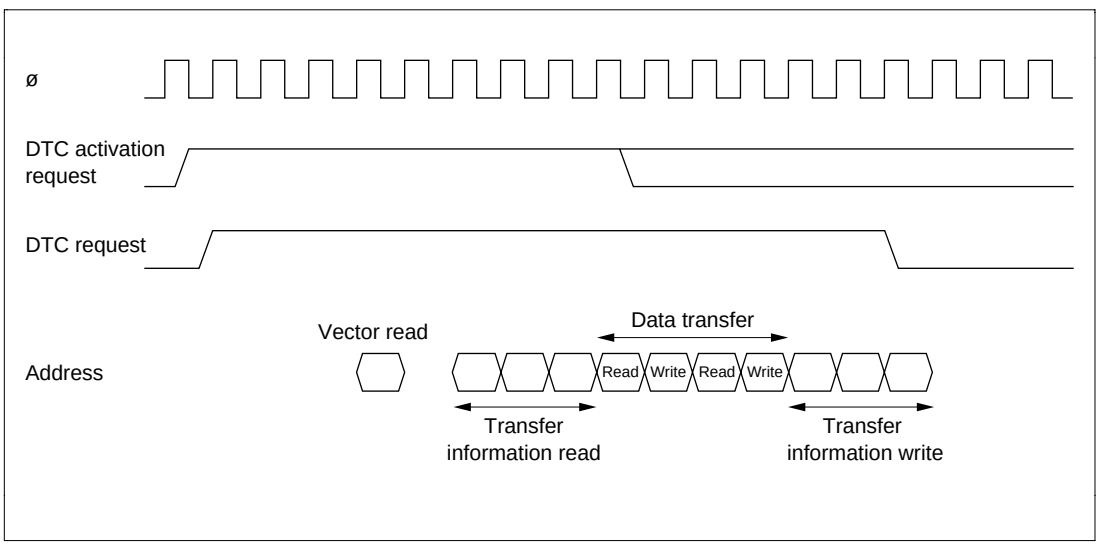



**Figure 8-11 DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2)**

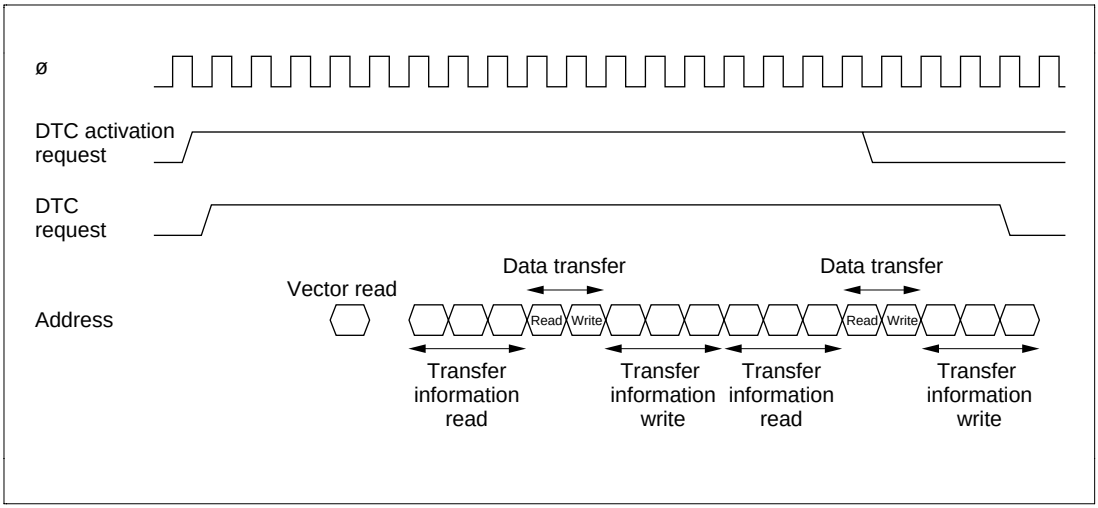



**Figure 8-12 DTC Operation Timing (Example of Chain Transfer)**

### 8.3.10 Number of DTC Execution States

Table 8-8 lists execution statuses for a single DTC data transfer, and table 8-9 shows the number of states required for each execution status.

**Table 8-8 DTC Execution Statuses**

| Mode | Vector Read I | Register Information Read/Write J | Data Read K | Data Write L | Internal Operations M |
|---|---|---|---|---|---|
| Normal | 1 | 6 | 1 | 1 | 3 |
| Repeat | 1 | 6 | 1 | 1 | 3 |
| Block transfer | 1 | 6 | N | N | 3 |

N: Block size (initial setting of CRAH and CRAL)

**Table 8-9 Number of States Required for Each Execution Status**

| Object to be Accessed | | | On-Chip RAM | On-Chip ROM | On-Chip I/O Registers | | External Devices | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bus width | | | 32 | 16 | 8 | 16 | 8 | | 16 | |
| Access states | | | 1 | 1 | 2 | 2 | 2 | 3 | 2 | 3 |
| Execution status | Vector read | $S_I$ | — | 1 | — | — | 4 | 6+2m | 2 | 3+m |
| | Register information read/write | $S_J$ | 1 | — | — | — | — | — | — | — |
| | Byte data read | $S_K$ | 1 | 1 | 2 | 2 | 2 | 3+m | 2 | 3+m |
| | Word data read | $S_K$ | 1 | 1 | 4 | 2 | 4 | 6+2m | 2 | 3+m |
| | Byte data write | $S_L$ | 1 | 1 | 2 | 2 | 2 | 3+m | 2 | 3+m |
| | Word data write | $S_L$ | 1 | 1 | 4 | 2 | 4 | 6+2m | 2 | 3+m |
| | Internal operation | $S_M$ | 1 | | | | | | | |

The number of execution states is calculated from the formula below. Note that Σ means the sum of all transfers activated by one activation event (the number in which the CHNE bit is set to 1, plus 1).

$$\text{Number of execution states} = I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L) + M \cdot S_M$$

For example, when the DTC vector address table is located in on-chip ROM, normal mode is set, and data is transferred from the on-chip ROM to an internal I/O register, the time required for the DTC operation is 13 states. The time from activation to the end of the data write is 10 states.

### 8.3.11 Procedures for Using DTC

**Activation by Interrupt:** The procedure for using the DTC with interrupt activation is as follows:

[1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.

[2] Set the start address of the register information in the DTC vector address.

[3] Set the corresponding bit in DTCER to 1.

[4] Set the enable bits for the interrupt sources to be used as the activation sources to 1. The DTC is activated when an interrupt used as an activation source is generated.

[5] After the end of one data transfer, or after the specified number of data transfers have ended, the DTCE bit is cleared to 0 and a CPU interrupt is requested. If the DTC is to continue transferring data, set the DTCE bit to 1.

**Activation by Software:** The procedure for using the DTC with software activation is as follows:

[1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.

[2] Set the start address of the register information in the DTC vector address.

[3] Check that the SWDTE bit is 0.

[4] Write 1 to SWDTE bit and the vector number to DTVECR.

[5] Check the vector number written to DTVECR.

[6] After the end of one data transfer, if the DISEL bit is 0 and a CPU interrupt is not requested, the SWDTE bit is cleared to 0. If the DTC is to continue transferring data, set the SWDTE bit to 1. When the DISEL bit is 1, or after the specified number of data transfers have ended, the SWDTE bit is held at 1 and a CPU interrupt is requested.

### 8.3.12 Examples of Use of the DTC

#### (1) Normal Mode

An example is shown in which the DTC is used to receive 128 bytes of data via the SCI.

[1] Set MRA to fixed source address (SM1 = SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), normal mode (MD1 = MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one data transfer by one interrupt (CHNE = 0, DISEL = 0). Set the SCI RDR address in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H'0080) in CRA. CRB can be set to any value.

[2] Set the start address of the register information at the DTC vector address.

[3] Set the corresponding bit in DTCER to 1.

[4] Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the reception complete (RXI) interrupt. Since the generation of a receive error during the SCI reception operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.

[5] Each time reception of one byte of data ends on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.

[6] When CRA becomes 0 after the 128 data transfers have ended, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. The interrupt handling routine should perform wrap-up processing.

#### (2) Chain Transfer

An example of DTC chain transfer is shown in which pulse output is performed using the PPG. Chain transfer can be used to perform pulse output data transfer and PPG output trigger cycle updating. Repeat mode transfer to the PPG's NDR is performed in the first half of the chain transfer, and normal mode transfer to the TPU's TGR in the second half. This is because clearing of the activation source and interrupt generation at the end of the specified number of transfers are limited to the second half of the chain transfer (transfer when CHNE = 0).

[1] Set transfer to the PPG's NDR. Set MRA to incrementing source address (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), repeat mode (MD1 = 0, MD0 = 1), and word size (Sz = 1). Set the source side as a repeat area (DTS = 1). Set MRB to chain mode (CHNE = 1, DISEL = 0). Set the data table start address in SAR, the NDRH address in DAR, and the data table size in CRAH and CRAL. CRB can be set to any value.

[2] Set transfer to the TPU's TGR. Set MRA to incrementing source address (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), normal mode (MD1 = MD0 = 0), and word size (Sz = 1). Set the data table start address in SAR, the TGRA address in DAR, and the data table size in CRA. CRB can be set to any value.

[3] Locate the TPU transfer register information consecutively after the NDR transfer register information.

[4] Set the start address of the NDR transfer register information at the DTC vector address.

[5] Set the bit corresponding to TGIA in DTCER to 1.

[6] Set TGRA as an output compare register (output disabled) with TIOR, and enable the TGIA interrupt with TIER.

[7] Set the initial output value in PODR, and the next output value in NDR. Set bits in DDR and NDER for which output is to be performed to 1. Using PCR, select the TPU compare match to be used as the output trigger.

[8] Set the CST bit in TSTR to 1, and start the TCNT count operation.

[9] Each time a TGRA compare match occurs, the next output value is transferred to NDR and the set value of the next output trigger period is transferred to TGRA. The activation source TGFA flag is cleared.

[10] When the specified number of transfers are completed (the TPU transfer CRA value is 0), the TGFA flag is held at 1, the DTCE bit is cleared to 0, and a TGIA interrupt request is sent to the CPU. The interrupt handling routine should perform wrap-up processing.

**(3) Software Activation**

An example is shown in which the DTC is used to transfer a block of 128 bytes of data by means of software activation. The transfer source address is H'1000 and the destination address is H'2000. The vector number is H'60, so the vector address is H'04C0.

[1] Set MRA to incrementing source address (SM1 = 1, SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), block transfer mode (MD1 = 1, MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one block transfer by one interrupt (CHNE = 0). Set the transfer source address (H'1000) in SAR, the destination address (H'2000) in DAR, and 128 (H'8080) in CRA. Set 1 (H'0001) in CRB.

[2] Set the start address of the register information at the DTC vector address (H'04C0).

[3] Check that the SWDTE bit in DTVECR is 0. Check that there is currently no transfer activated by software.

[4] Write 1 to the SWDTE bit and the vector number (H'60) to DTVECR. The write data is H'E0.

[5] Read DTVECR again and check that it is set to the vector number (H'60). If it is not, this indicates that the write failed. This is presumably because an interrupt occurred between steps 3 and 4 and led to a different software activation. To activate this transfer, go back to step 3.

[6] If the write was successful, the DTC is activated and a block of 128 bytes of data is transferred.

[7] After the transfer, an SWDTEND interrupt occurs. The interrupt handling routine should clear the SWDTE bit to 0 and perform other wrap-up processing.

## 8.4 Interrupts

An interrupt request is issued to the CPU when the DTC finishes the specified number of data transfers, or a data transfer for which the DISEL bit was set to 1. In the case of interrupt activation, the interrupt set as the activation source is generated. These interrupts to the CPU are subject to CPU mask level and interrupt controller priority level control.

In the case of activation by software, a software activated data transfer end interrupt (SWDTEND) is generated.

When the DISEL bit is 1 and one data transfer has ended, or the specified number of transfers have ended, after data transfer ends, the SWDTE bit is held at 1 and an SWDTEND interrupt is generated. The interrupt handling routine should clear the SWDTE bit to 0.

When the DTC is activated by software, an SWDTEND interrupt is not generated during a data transfer wait or during data transfer even if the SWDTE bit is set to 1.

## 8.5 Usage Notes

**Module Stop:** When the MSTP14 bit in MSTPCR is set to 1, the DTC clock stops, and the DTC enters the module stop state. However, 1 cannot be written in the MSTP14 bit while the DTC is operating.

**On-Chip RAM:** The MRA, MRB, SAR, DAR, CRA, and CRB registers are all located in on-chip RAM. When the DTC is used, the RAME bit in SYSCR must not be cleared to 0.

**DMAC Transfer End Interrupt:** When DTC transfer is activated by a DMAC transfer end interrupt, regardless of the transfer counter and DISEL bit, the DMAC's DTE bit is not subject to DTC control, and the write data has priority. Consequently, an interrupt request is not sent to the CPU when the DTC transfer counter reaches 0.

# Section 9 I/O Ports

## 9.1 Overview

The H8S/2655 Series has 12 I/O ports (ports 1, 2, 3, 5, 6, and A to G), and one input-only port (port 4).

Table 9-1 summarizes the port functions. The pins of each port also have other functions.

Each port includes a data direction register (DDR) that controls input/output (not provided for the input-only port), a data register (DR) that stores output data, and a port register (PORT) used to read the pin states.

Ports A to E have a built-in MOS input pull-up function, and in addition to DR and DDR, have a MOS input pull-up control register (PCR) to control the on/off state of MOS input pull-up.

Ports 3 and A include an open-drain control register (ODR) that controls the on/off state of the output buffer PMOS.

Ports A to E can drive a single TTL load and 90 pF capacitive load, and ports 1, 2, 3, 5, 6, F, and G can drive a single TTL load and 30 pF capacitive load. All the I/O ports can drive a Darlington transistor when in output mode. Ports 1, and A to C can drive an LED (10 mA sink current).

Port 2, and ports $6_4$ to $6_7$ and $A_4$ to $A_7$, are Schmitt-triggered inputs.

For block diagrams of the ports see appendix C, I/O Port Block Diagrams.

| Port | Description | Pins | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|---|
| Port 1 | • 8-bit I/O port | $P1_7/PO_{15}/TIOCB_2/TCLKD$<br>$P1_6/PO_{14}/TIOCA_2$<br>$P1_5/PO_{13}/TIOCB_1/TCLKC$<br>$P1_4/PO_{12}/TIOCA_1$<br>$P1_3/PO_{11}/TIOCD_0/TCLKB$<br>$P1_2/PO_{10}/TIOCC_0/TCLKA$<br>$P1_1/PO_9/TIOCB_0/\overline{DACK}_1$<br>$P1_0/PO_8/TIOCA_0/\overline{DACK}_0$ | 8-bit I/O port also functioning as DMA controller output pins ($\overline{DACK}_0$ and $\overline{DACK}_1$), TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, $TIOCA_0$, $TIOCB_0$, $TIOCC_0$, $TIOCD_0$, $TIOCA_1$, $TIOCB_1$, $TIOCA_2$, $TIOCB_2$) and PPG output pins ($PO_{15}$ to $PO_8$) | | | | | | |
| Port 2 | • 8-bit I/O port<br>• Schmitt-triggered input | $P2_7/PO_7/TIOCB_5/TMO_1$<br>$P2_6/PO_6/TIOCA_5/TMO_0$<br>$P2_5/PO_5/TIOCB_4/TMCI_1$<br>$P2_4/PO_4/TIOCA_4/TMRI_1$<br>$P2_3/PO_3/TIOCD_3/TMCI_0$<br>$P2_2/PO_2/TIOCC_3/TMRI_0$<br>$P2_1/PO_1/TIOCB_3$<br>$P2_0/PO_0/TIOCA_3$ | 8-bit I/O port also functioning as TPU I/O pins ($TIOCA_3$, $TIOCB_3$, $TIOCC_3$, $TIOCD_3$, $TIOCA_4$, $TIOCB_4$, $TIOCA_5$, $TIOCB_5$), 8-bit timer (channels 0 and 1) I/O pins ($TMRI_0$, $TMCI_0$, $TMO_0$, $TMRI_1$, $TMCI_1$, $TMO_1$) and PPG output pins ($PO_7$ to $PO_0$) | | | | | | |
| Port 3 | • 6-bit I/O port<br>• Open-drain output capability | $P3_5/SCK_1$<br>$P3_4/SCK_0$<br>$P3_3/RxD_1$<br>$P3_2/RxD_0$<br>$P3_1/TxD_1$<br>$P3_0/TxD_0$ | 6-bit I/O port also functioning as SCI (channels 0 and 1) I/O pins ($TxD_0$, $RxD_0$, $SCK_0$, $TxD_1$, $RxD_1$, $SCK_1$) | | | | | | |
| Port 4 | • 8-bit input port | $P4_7/AN_7/DA_1$<br>$P4_6/AN_6/DA_0$<br>$P4_5/AN_5$<br>$P4_4/AN_4$<br>$P4_3/AN_3$<br>$P4_2/AN_2$<br>$P4_1/AN_1$<br>$P4_0/AN_0$ | 8-bit input port also functioning as A/D converter analog inputs ($AN_7$ to $AN_0$) and D/A converter analog outputs ($DA_1$ and $DA_0$) | | | | | | |

| Port | Description | Pins | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|---|
| Port 5 | • 4-bit I/O port | $P5_3/\overline{ADTRG}$<br>$P5_2/SCK_2$<br>$P5_1/RxD_2$<br>$P5_0/TxD_2$ | 4-bit I/O port also functioning as SCI (channel 2) I/O pins ($TxD_2$, $RxD_2$, $SCK_2$) and A/D converter input pin ($\overline{ADTRG}$) | | | | | | |
| Port 6 | • 8-bit I/O port<br>• Schmitt-triggered input ($P6_4$ to $P6_7$) | $P6_7/\overline{IRQ_3}/\overline{CS_7}$<br>$P6_6/\overline{IRQ_2}/\overline{CS_6}$<br>$P6_5/\overline{IRQ_1}$<br>$P6_4/\overline{IRQ_0}$<br>$P6_3/\overline{TEND_1}$<br>$P6_2/\overline{DREQ_1}$<br>$P6_1/\overline{TEND_0}/\overline{CS_5}$<br>$P6_0/\overline{DREQ_0}/\overline{CS_4}$ | 8-bit I/O port also functioning as DMA controller I/O pins ($\overline{DREQ_0}$, $\overline{TEND_0}$, $\overline{DREQ_1}$, $\overline{TEND_1}$) and interrupt input pins ($\overline{IRQ_0}$ to $\overline{IRQ_3}$) | | | 8-bit I/O port also functioning as DMA controller I/O pins ($\overline{DREQ_0}$, $\overline{TEND_0}$, $\overline{DREQ_1}$, $\overline{TEND_1}$), bus control output pins ($\overline{CS_4}$ to $\overline{CS_7}$), and interrupt input pins ($\overline{IRQ_0}$ to $\overline{IRQ_3}$) | | | 8-bit I/O port also functioning as interrupt input pins ($\overline{IRQ_0}$ to $\overline{IRQ_3}$) |

| Port | Description | Pins | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|---|
| Port A | • 8-bit I/O port<br>• Built-in MOS input pull-up<br>• Open-drain output capability<br>• Schmitt-triggered input ($PA_4$ to $PA_7$) | $PA_7/A_{23}/\overline{IRQ_7}$<br>$PA_6/A_{22}/\overline{IRQ_6}$<br>$PA_5/A_{21}/\overline{IRQ_5}$ | Dual function as I/O ports and interrupt input pins ($\overline{IRQ_7}$ to $\overline{IRQ_4}$) | | | When DDR = 0 (after reset): dual function as input ports and interrupt input pins ($\overline{IRQ_7}$ to $\overline{IRQ_5}$)<br>When DDR = 1: address output | | When DDR = 0 (after reset): dual function as input ports and interrupt input pins ($\overline{IRQ_7}$ to $\overline{IRQ_4}$)<br>When DDR = 1: address output | Dual function as I/O ports and interrupt input pins ($\overline{IRQ_7}$ to $\overline{IRQ_4}$) |
| | | $PA_4/A_{20}/\overline{IRQ_4}$ | | | | Address output | | | |
| | | $PA_3/A_{19}$ to $PA_0/A_{16}$ | I/O ports | | | Address output | | When DDR = 0 (after reset): input ports<br>When DDR = 1: address output | I/O ports |
| Port B | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PB_7/A_{15}$ to $PB_0/A_8$ | Address output | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port | Address output | | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port |

| Port | Description | Pins | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|---|
| Port C | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PC_7/A_7$ to $PC_0/A_0$ | Address output | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port | Address output | | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port |
| Port D | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PD_7/D_{15}$ to $PD_0/D_8$ | Data bus input/output | | I/O port | Data bus input/output | | | I/O port |
| Port E | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PE_7/D_7$ to $PE_0/D_0$ | In 8-bit bus mode: I/O port<br>In 16-bit bus mode: data bus input/output | | I/O port | In 8-bit bus mode: I/O port<br>In 16-bit bus mode: data bus input/output | | | I/O port |
| Port F | • 8-bit I/O port | $PF_7/ø$ | When DDR = 0: input port<br>When DDR = 1 (after reset): ø output | | When DDR = 0 (after reset): input port<br>When DDR = 1: ø output | When DDR = 0: input port<br>When DDR = 1 (after reset): ø output | | | When DDR = 0 (after reset): input port<br>When DDR = 1: ø output |
| | | $PF_6/\overline{AS}$<br>$PF_5/\overline{RD}$<br>$PF_4/\overline{HWR}$<br>$PF_3/\overline{LWR}$ | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ output | | I/O port | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ output | | | I/O port |

| Port | Description | Pins | Mode 1 | Mode 2 | Mode 3 | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|---|---|---|---|
| Port F | • 8-bit I/O port | $PF_2/\overline{LCAS}/\overline{WAIT}/\overline{BREQO}$ | When WAITE = 0 and BREQOE = 0 (after reset): I/O port<br>When WAITE = 1 and BREQOE = 0: $\overline{WAIT}$ input<br>When WAITE = 0 and BREQOE = 1: $\overline{BREQO}$ input | | I/O port | When WAITE = 0 and BREQOE = 0 (after reset): I/O port<br>When WAITE = 1 and BREQOE = 0: $\overline{WAIT}$ input<br>When WAITE = 0 and BREQOE = 1: $\overline{BREQO}$ output<br>When RMTS2 to RMTS0= B'001 to B'011, CW2= 0, and LCASS= 0: $\overline{LCAS}$ output | | | I/O port |
| | | $PF_1/\overline{BACK}$<br>$PF_0/\overline{BREQ}$ | When BRLE = 0 (after reset): I/O port<br>When BRLE = 1: $\overline{BREQ}$ input, $\overline{BACK}$ output | | | When BRLE = 0 (after reset): I/O port<br>When BRLE = 1: $\overline{BREQ}$ input, $\overline{BACK}$ output | | | |
| Port G | • 5-bit I/O port | $PG_4/\overline{CS_0}$ | When DDR= 0*1: input port<br>When DDR= 1*2: $\overline{CS_0}$ output | | I/O port | When DDR = 0*1: input port<br>When DDR = 1*2: $\overline{CS_0}$ output | | | I/O port |
| | | $PG_3/\overline{CS_1}$<br>$PG_2/\overline{CS_2}$<br>$PG_1/\overline{CS_3}$ | I/O port | | | When DDR = 0 (after reset): input port<br>When DDR = 1: $\overline{CS_1}$, $\overline{CS_2}$, $CS_3$ output | | | |
| | | $PG_0/\overline{CAS}/\overline{OE}$ | | | | DRAM space set: $\overline{CAS}$ output<br>PSRAM space set: $\overline{OE}$ output<br>Otherwise (after reset): I/O port | | | |

Notes: 1. After a reset in mode 2 or 6
2. After a reset in mode 1, 4 or 5

## 9.2 Port 1

### 9.2.1 Overview

Port 1 is an 8-bit I/O port. Port 1 pins also function as PPG output pins ($PO_{15}$ to $PO_8$), TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, $TIOCA_0$, $TIOCB_0$, $TIOCC_0$, $TIOCD_0$, $TIOCA_1$, $TIOCB_1$, $TIOCA_2$, and $TIOCB_2$), and DMAC output pins ($\overline{DACK}_0$ and $\overline{DACK}_1$). Port 1 pin functions are the same in all operating modes.

Figure 9-1 shows the port 1 pin configuration.

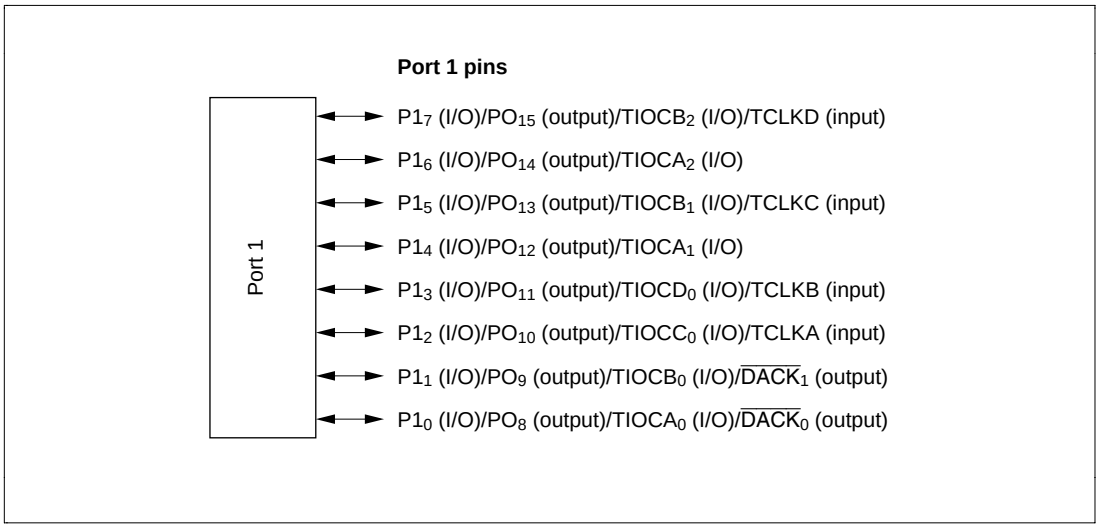



**Figure 9-1 Port 1 Pin Functions**

### 9.2.2 Register Configuration

Table 9-2 shows the port 1 register configuration.

**Table 9-2 Port 1 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 1 data direction register | P1DDR | W | H'00 | H'FEB0 |
| Port 1 data register | P1DR | R/W | H'00 | H'FF60 |
| Port 1 register | PORT1 | R | Undefined | H'FF50 |

Note: * Lower 16 bits of the address.

**Port 1 Data Direction Register (P1DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P1DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 1. P1DDR cannot be read; if it is, an undefined value will be read.

Setting a P1DDR bit to 1 makes the corresponding port 1 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P1DDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the PPG, TPU, and DMAC are initialized by a manual reset, the pin states are determined by the P1DDR and P1DR specifications.

**Port 1 Data Register (P1DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

P1DR is an 8-bit readable/writable register that stores output data for the port 1 pins ($P1_7$ to $P1_0$).

P1DR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 1 Register (PORT1)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P1_7$ to $P1_0$.

PORT1 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 1 pins ($P1_7$ to $P1_0$) must always be performed on P1DR.

If a port 1 read is performed while P1DDR bits are set to 1, the P1DR values are read. If a port 1 read is performed while P1DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT1 contents are determined by the pin states, as P1DDR and P1DR are initialized. PORT1 retains its prior state after a manual reset, and in software standby mode.

### 9.2.3 Pin Functions

Port 1 pins also function as PPG output pins ($PO_{15}$ to $PO_8$), TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, $TIOCA_0$, $TIOCB_0$, $TIOCC_0$, $TIOCD_0$, $TIOCA_1$, $TIOCB_1$, $TIOCA_2$, and $TIOCB_2$), and DMAC output pins ($\overline{DACK_0}$ and $\overline{DACK_1}$). Port 1 pin functions are shown in table 9-3.

**Table 9-3 Port 1 Pin Functions**

<table>
<tr><th>Pin</th><th>Selection Method and Pin Functions</th></tr>
<tr><td>$P1_7$/$PO_{15}$/$TIOCB_2$/<br>TCLKD</td><td>The pin function is switched as shown below according to the combination of the TPU channel 2 setting by bits MD3 to MD0 in TMDR2, bits IOB3 to IOB0 in TIOR2, bits CCLR1 and CCLR0 in TCR2, bits TPSC2 to TPSC0 in TCR0 and TCR5, bit NDER15 in NDERH, and bit P17DDR.
<table>
<tr><td>TPU Channel 2 Setting</td><td>Table Below (1)</td><td colspan="3">Table Below (2)</td></tr>
<tr><td>P17DDR</td><td>—</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>NDER15</td><td>—</td><td>—</td><td>0</td><td>1</td></tr>
<tr><td rowspan="2">Pin function</td><td rowspan="2">$TIOCB_2$ output</td><td>$P1_7$ input</td><td>$P1_7$ output</td><td>$PO_{15}$ output</td></tr>
<tr><td colspan="3">$TIOCB_2$ input *1</td></tr>
<tr><td colspan="4">TCLKD input *2</td></tr>
</table>
Notes: 1. $TIOCB_2$ input when MD3 to MD0 = B'0000, B'01xx, and IOB3 = 1.<br>2. TCLKD input when the setting for either TCR0 or TCR5 is: TPSC2 to TPSC0 = B'111.<br>TCLKD input when channels 2 and 4 are set to phase counting mode.
<table>
<tr><td>TPU Channel 2 Setting</td><td>(2)</td><td>(1)</td><td>(2)</td><td>(2)</td><td>(1)</td><td>(2)</td></tr>
<tr><td>MD3 to MD0</td><td colspan="2">B'0000, B'01xx</td><td>B'0010</td><td colspan="3">B'0011</td></tr>
<tr><td>IOB3 to IOB0</td><td>B'0000<br>B'0100<br>B'1xxx</td><td>B'0001 to B'0011<br>B'0101 to B'0111</td><td>—</td><td>B'xx00</td><td colspan="2">Other than B'xx00</td></tr>
<tr><td>CCLR1, CCLR0</td><td>—</td><td>—</td><td>—</td><td>—</td><td>Other than B'10</td><td>B'10</td></tr>
<tr><td>Output function</td><td>—</td><td>Output compare output</td><td>—</td><td>—</td><td>PWM mode 2 output</td><td>—</td></tr>
</table>
x: Don’t care</td></tr>
</table>

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_6/PO_{14}/TIOCA_2$ | The pin function is switched as shown below according to the combination of the TPU channel 2 setting by bits MD3 to MD0 in TMDR2, bits IOA3 to IOA0 in TIOR2, bits CCLR1 and CCLR0 in TCR2, bit NDER14 in NDERH, and bit P16DDR. |

| TPU Channel 2 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P16DDR | — | 0 | 1 | 1 |
| NDER14 | — | — | 0 | 1 |
| Pin function | $TIOCA_2$ output | $P1_6$ input | $P1_6$ output | $PO_{14}$ output |
| | | $TIOCA_2$ input *1 | | |

Note: 1. $TIOCA_2$ input when MD3 to MD0 = B'0000, B'01xx, and IOA3 = 1.

| TPU Channel 2 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output *2 | PWM mode 2 output | — |

x: Don't care

Note: 2. $TIOCB_2$ output is disabled.

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_5/PO_{13}/TIOCB_1/TCLKC$ | The pin function is switched as shown below according to the combination of the TPU channel 1 setting by bits MD3 to MD0 in TMDR1, bits IOB3 to IOB0 in TIOR1, bits CCLR1 and CCLR0 in TCR1, bits TPSC2 to TPSC0 in TCR0, TCR2, TCR4, and TCR5, bit NDER13 in NDERH, and bit P15DDR. |

| TPU Channel 1 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P15DDR | — | 0 | 1 | 1 |
| NDER13 | — | — | 0 | 1 |
| Pin function | $TIOCB_1$ output | $P1_5$ input | $P1_5$ output | $PO_{13}$ output |
| | | $TIOCB_1$ input *1 | | |
| | TCLKC input *2 | | | |

Notes: 1. $TIOCB_1$ input when MD3 to MD0 = B'0000, B'01xx and IOB3 to IOB0 = B'10xx.
2. TCLKC input when the setting for either TCR0 or TCR2 is: TPSC2 to TPSC0 = B'110; or when the setting for either TCR4 or TCR5 is TPSC2 to TPSC0 = B'101.
TCLKC input when channels 2 and 4 are set to phase counting mode.

| TPU Channel 1 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_4/PO_{12}/TIOCA_1$ | The pin function is switched as shown below according to the combination of the TPU channel 1 setting by bits MD3 to MD0 in TMDR1, bits IOA3 to IOA0 in TIOR1, bits CCLR1 and CCLR0 in TCR1, bit NDER12 in NDERH, and bit P14DDR. |

| TPU Channel 1 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P14DDR | — | 0 | 1 | 1 |
| NDER12 | — | — | 0 | 1 |
| Pin function | $TIOCA_1$ output | $P1_4$ input | $P1_4$ output | $PO_{12}$ output |
| | | $TIOCA_1$ input *1 | | |

Note: 1. $TIOCA_1$ input when MD3 to MD0 = B'0000, B'01xx, IOA3 to IOA0 = B'10xx.

| TPU Channel 1 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Note: 2. $TIOCB_1$ output is disabled.

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_3/PO_{11}/TIOCD_0/$ TCLKB | The pin function is switched as shown below according to the combination of the TPU channel 0 setting by bits MD3 to MD0 in TMDR0, bits IOD3 to IOD0 in TIOR0L, bits CCLR2 to CCLR0 in TCR0, bits TPSC2 to TPSC0 in TCR0 to TCR2, bit NDER11 in NDERH, and bit P13DDR. |

| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P13DDR | — | 0 | 1 | 1 |
| NDER11 | — | — | 0 | 1 |
| Pin function | $TIOCD_0$ output | $P1_3$ input | $P1_3$ output | $PO_{11}$ output |
| | | $TIOCD_0$ input *1 | | |
| | TCLKB input *2 | | | |

Notes: 1. $TIOCD_0$ input when MD3 to MD0 = B'0000, IOD3 to IOD0 =B'10xx.
2. TCLKB input when the setting for TCR0 to TCR2 is: TPSC2 to TPSC0 = B'101;
TCLKB input when channels 1 and 5 are set to phase counting mode.

| TPU Channel 0 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOD3 to IOD0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'110 | B'110 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_2/PO_{10}/TIOCC_0$/ TCLKA | The pin function is switched as shown below according to the combination of the TPU channel 0 setting by bits MD3 to MD0 in TMDR0, bits IOC3 to IOC0 in TIOR0L, bits CCLR2 to CCLR0 in TCR0, bits TPSC2 to TPSC0 in TCR0 to TCR5, bit NDER10 in NDERH, and bit P12DDR. |

| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P12DDR | — | 0 | 1 | 1 |
| NDER10 | — | — | 0 | 1 |
| Pin function | $TIOCC_0$ output | $P1_2$ input | $P1_2$ output | $PO_{10}$ output |
| | | $TIOCC_0$ input *1 | | |
| | TCLKA input *2 | | | |

Notes: 1. $TIOCC_0$ input when MD3 to MD0 = B'0000, and IOC3 to IOC0 = B'10xx.
2. TCLKA input when the setting for TCR0 to TCR5 is: TPSC2 to TPSC0 = B'100;
TCLKA input when channels 1 and 5 are set to phase counting mode.

| TPU Channel 0 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOC3 to IOC0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'101 | B'101 |
| Output function | — | Output compare output | — | PWM mode 1 output*3 | PWM mode 2 output | — |

x: Don't care

Note: 3. $TIOCD_0$ output is disabled.
When BFA = 1 or BFB = 1 in TMDR0, output is disabled and setting (2) applies.

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_1/PO_9/TIOCB_0/\overline{DACK}_1$ | The pin function is switched as shown below according to the combination of the TPU channel 0 setting by bits MD3 to MD0 in TMDR0, bits IOB3 to IOB0 in TIOR0H, bits CCLR2 to CCLR0 in TCR0, bit NDER9 in NDERH, bit SAE1 in DMABCRH, and bit P11DDR. |

| SAE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | | |
| P11DDR | — | 0 | 1 | 1 | — |
| NDER9 | — | — | 0 | 1 | — |
| Pin function | $TIOCB_0$ output | $P1_1$ input | $P1_1$ output | $PO_9$ output | $\overline{DACK}_1$ output |
| | | TIOCB0 input *1 | | | |

Note: 1. $TIOCB_0$ input when MD3 to MD0 = B'0000, and IOB3 to IOB0 = B'10xx.

| TPU Channel 0 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'010 | B'010 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_0/PO_8/TIOCA_0/\overline{DACK}_0$ | The pin function is switched as shown below according to the combination of the TPU channel 0 setting by bits MD3 to MD0 in TMDR0, bits IOA3 to IOA0 in TIOR0H, bits CCLR2 to CCLR0 in TCR0, bit NDER8 in NDERH, bit SAE0 in DMABCRH, and bit P10DDR. |

| SAE0 | 0 | | | | 1 |
|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | | — |
| P10DDR | — | 0 | 1 | 1 | — |
| NDER8 | — | — | 0 | 1 | — |
| Pin function | $TIOCA_0$ output | $P1_0$ input | $P1_0$ output | $PO_8$ output | $\overline{DACK}_0$ output |
| | | $TIOCA_0$ input *1 | | | |

Note: 1. $TIOCA_0$ input when MD3 to MD0 = B'0000, and IOA3 to IOA0 = B'10xx.

| TPU Channel 0 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'001 | B'001 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Note: 2. $TIOCB_0$ output is disabled.

## 9.3 Port 2

### 9.3.1 Overview

Port 2 is an 8-bit I/O port. Port 2 pins also function as PPG output pins ($PO_7$ to $PO_0$), TPU I/O pins ($TIOCA_3$, $TIOCB_3$, $TIOCC_3$, $TIOCD_3$, $TIOCA_4$, $TIOCB_4$, $TIOCA_5$, and $TIOCB_5$), and 8-bit timer I/O pins ($TMRI_0$, $TMCI_0$, $TMO_0$, $TMRI_1$, $TMCI_1$, and $TMO_1$). Port 2 pin functions are the same in all operating modes. Port 2 uses Schmitt-triggered input.

Figure 9-2 shows the port 2 pin configuration.

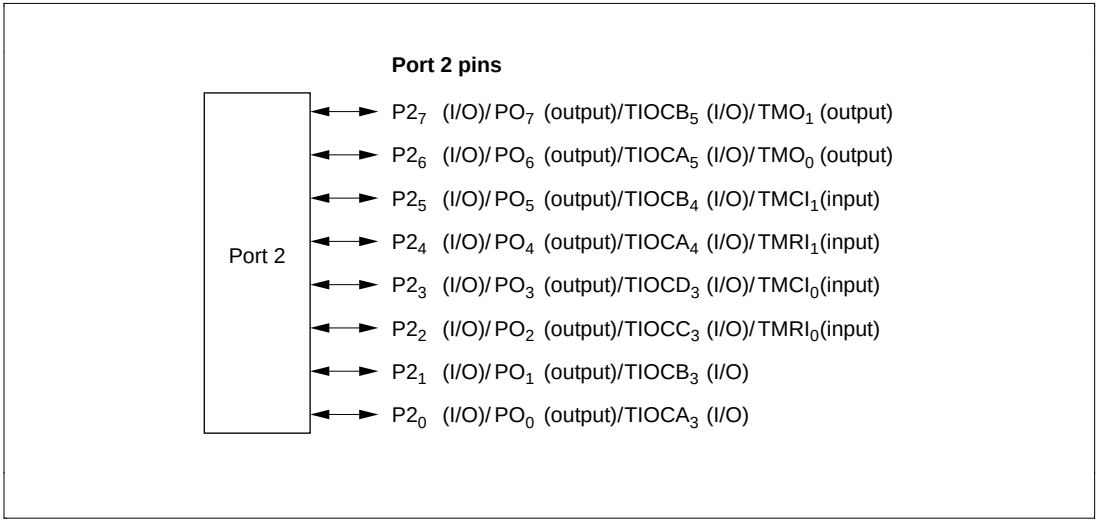



**Figure 9-2 Port 2 Pin Functions**

### 9.3.2 Register Configuration

Table 9-4 shows the port 2 register configuration.

**Table 9-4 Port 2 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 2 data direction register | P2DDR | W | H'00 | H'FEB1 |
| Port 2 data register | P2DR | R/W | H'00 | H'FF61 |
| Port 2 register | PORT2 | R | Undefined | H'FF51 |

Note: * Lower 16 bits of the address.

**Port 2 Data Direction Register (P2DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P2DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 2. P2DDR cannot be read; if it is, an undefined value will be read.

Setting a P2DDR bit to 1 makes the corresponding port 2 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P2DDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the PPG, TPU, and 8-bit timer are initialized by a manual reset, the pin states are determined by the P2DDR and P2DR specifications.

**Port 2 Data Register (P2DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

P2DR is an 8-bit readable/writable register that stores output data for the port 2 pins ($P2_7$ to $P2_0$).

P2DR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 2 Register (PORT2)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P2_7$ to $P2_0$.

PORT2 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 2 pins ($P2_7$ to $P2_0$) must always be performed on P2DR.

If a port 2 read is performed while P2DDR bits are set to 1, the P2DR values are read. If a port 2 read is performed while P2DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT2 contents are determined by the pin states, as P2DDR and P2DR are initialized. PORT2 retains its prior state after a manual reset, and in software standby mode.

### 9.3.3 Pin Functions

Port 2 pins also function as PPG output pins ($PO_7$ to $PO_0$), TPU I/O pins ($TIOCA_3$, $TIOCB_3$, $TIOCC_3$, $TIOCD_3$, $TIOCA_4$, $TIOCB_4$, $TIOCA_5$, and $TIOCB_5$), and 8-bit timer I/O pins ($TMRI_0$, $TMCI_0$, $TMO_0$, $TMRI_1$, $TMCI_1$, and $TMO_1$). Port 2 pin functions are shown in table 9-5.

**Table 9-5 Port 2 Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_7/PO_7/TIOCB_5/TMO_1$ | The pin function is switched as shown below according to the combination of the TPU channel 5 setting by bits MD3 to MD0 in TMDR5, bits IOB3 to IOB0 in TIOR5, bits CCLR1 and CCLR0 in TCR5, bit NDER7 in NDERL, bits OS3 to OS0 in TCSR1, and bit P27DDR. |

| OS3 to OS0 | All 0 | | | | Any 1 |
|---|---|---|---|---|---|
| TPU Channel 5 Setting | Table Below (1) | Table Below (2) | | | — |
| P27DDR | — | 0 | 1 | 1 | — |
| NDER7 | — | — | 0 | 1 | — |
| Pin function | $TIOCB_5$ output | $P2_7$ input | $P2_7$ output | $PO_7$ output | $TMO_1$ output |
| | | $TIOCB_5$ input * | | | |

Note: * $TIOCB_5$ input when MD3 to MD0 = B'0000, B'01xx, and IOB3 = 1.

| TPU Channel 5 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_6/PO_6/TIOCA_5/TMO_0$ | The pin function is switched as shown below according to the combination of the TPU channel 5 setting by bits MD3 to MD0 in TMDR5, bits IOA3 to IOA0 in TIOR5, bits CCLR1 and CCLR0 in TCR5, bit NDER6 in NDERL, bits OS3 to OS0 in TCSR0, and bit P26DDR. |

| OS3 to OS0 | All 0 | | | | Any 1 |
|---|---|---|---|---|---|
| TPU Channel 5 Setting | Table Below (1) | Table Below (2) | | | — |
| P26DDR | — | 0 | 1 | 1 | — |
| NDER6 | — | — | 0 | 1 | — |
| Pin function | $TIOCA_5$ output | $P2_6$ input | $P2_6$ output | $PO_6$ output | $TMO_0$ output |
| | | $TIOCA_5$ input *1 | | | |

Note: 1. $TIOCA_5$ input when MD3 to MD0 = B'0000, B'01xx, and IOA3 = 1.

| TPU Channel 5 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Note: 2. $TIOCB_5$ output is disabled.

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_5/PO_5/TIOCB_4/TMCI_1$ | This pin is used as the 8-bit timer external clock input pin when external clock is selected with bits CKS2 to CKS0 in TCR1.<br>The pin function is switched as shown below according to the combination of the TPU channel 4 setting by bits MD3 to MD0 in TMDR4 and bits IOB3 to IOB0 in TIOR4, bits CCLR1 and CCLR0 in TCR4, bit NDER5 in NDERL, and bit P25DDR. |

| TPU Channel 4 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P25DDR | — | 0 | 1 | 1 |
| NDER5 | — | — | 0 | 1 |
| Pin function | $TIOCB_4$ output | $P2_5$ input | $P2_5$ output | $PO_5$ output |
| | | $TIOCB_4$ input *1 | | |
| | $TMCI_1$ input | | | |

Note: 1. $TIOCB_4$ input when MD3 to MD0 = B'0000, B'01xx, and IOB3 to IOB0 = B'10xx.

| TPU Channel 4 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_4$/$PO_4$/$TIOCA_4$/ $TMRI_1$ | This pin is used as the 8-bit timer counter reset pin when bits CCLR1 and CCLR0 in TCR1 are both set to 1.<br>The pin function is switched as shown below according to the combination of the TPU channel 4 setting by bits MD3 to MD0 in TMDR4, bits IOA3 to IOA0 in TIOR4, bits CCLR1 and CCLR0 in TCR4, bit NDER4 in NDERL, and bit P24DDR. |

| TPU Channel 4 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P24DDR | — | 0 | 1 | 1 |
| NDER4 | — | — | 0 | 1 |
| Pin function | $TIOCA_4$ output | $P2_4$ input | $P2_4$ output | $PO_4$ output |
| | | $TIOCA_4$ input *1 | | |
| | $TMRI_1$ input | | | |

Note: 1. $TIOCA_4$ input when MD3 to MD0 = B'0000, B'01xx, and IOA3 to IOA0 = B'10xx.

| TPU Channel 4 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Notes: 2. $TIOCB_4$ output is disabled.

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_3/PO_3/TIOCD_3/TMCI_0$ | This pin is used as the 8-bit timer external clock input pin when external clock is selected with bits CKS2 to CKS0 in TCR0.<br>The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOD3 to IOD0 in TIOR3L, bits CCLR2 to CCLR0 in TCR3, bit NDER3 in NDERL, and bit P23DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P23DDR | — | 0 | 1 | 1 |
| NDER3 | — | — | 0 | 1 |
| Pin function | $TIOCD_3$ output | $P2_3$ input | $P2_3$ output | $PO_3$ output |
| | | $TIOCD_3$ input *1 | | |
| | $TMCI_0$ input | | | |

Note: 1. $TIOCD_3$ input when MD3 to MD0 = B'0000, and IOD3 to IOD0 = B'10xx.

| TPU Channel 3 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOD3 to IOD0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'110 | B'110 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_2/PO_2/TIOCC_3/TMCI_0$ | This pin is used as the 8-bit timer counter reset pin when bits CCLR1 and CCLR0 in TCR0 are both set to 1.<br>The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOC3 to IOC0 in TIOR3L, bits CCLR2 to CCLR0 in TCR3, bit NDER2 in NDERL, and bit P22DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P22DDR | — | 0 | 1 | 1 |
| NDER2 | — | — | 0 | 1 |
| Pin function | $TIOCC_3$ output | $P2_2$ input | $P2_2$ output | $PO_2$ output |
| | | $TIOCC_3$ input *1 | | |
| | $TMRI_0$ input | | | |

Note: 1. $TIOCC_3$ input when MD3 to MD0 = B'0000, and IOC3 to IOC0 = B'10xx.

| TPU Channel 3 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOC3 to IOC0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'101 | B'101 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Note: 2. $TIOCD_3$ output is disabled.
When BFA = 1 or BFB = 1 in TMDR3, output is disabled and setting (2) applies.

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_1/PO_1/TIOCB_3$ | The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOB3 to IOB0 in TIOR3H, bits CCLR2 to CCLR0 in TCR3, bit NDER1 in NDERL, and bit P21DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P21DDR | — | 0 | 1 | 1 |
| NDER1 | — | — | 0 | 1 |
| Pin function | $TIOCB_3$ output | $P2_1$ input | $P2_1$ output | $PO_1$ output |
| | | $TIOCB_3$ input *1 | | |

Note: 1. $TIOCB_3$ input when MD3 to MD0 = B'0000, and IOB3 to IOB0 = B'10xx.

| TPU Channel 3 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'010 | B'010 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

**Table 9-5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_0/PO_0/TIOCA_3$ | The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOA3 to IOA0 in TIOR3H, bits CCLR2 to CCLR0 in TCR3, bit NDER0 in NDERL, and bit P20DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | | |
|---|---|---|---|---|
| P20DDR | — | 0 | 1 | 1 |
| NDER0 | — | — | 0 | 1 |
| Pin function | $TIOCA_3$ output | $P2_0$ input | $P2_0$ output | $PO_0$ output |
| | | $TIOCA_3$ input *1 | | |

Note: 1. $TIOCA_3$ input when MD3 to MD0 = B'0000, and IOA3 to IOA0 = B'10xx.

| TPU Channel 3 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'001 | B'001 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Notes: 2. $TIOCB_3$ output is disabled.

## 9.4 Port 3

### 9.4.1 Overview

Port 3 is a 6-bit I/O port. Port 3 pins also function as SCI I/O pins ($TxD_0$, $RxD_0$, $SCK_0$, $TxD_1$, $RxD_1$, and $SCK_1$). Port 3 pin functions are the same in all operating modes.

Figure 9-3 shows the port 3 pin configuration.

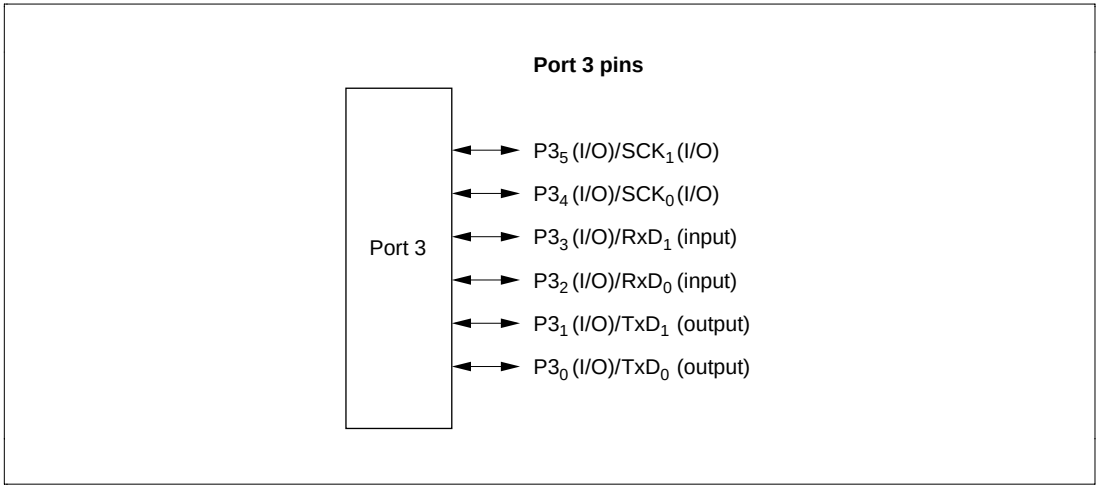



**Figure 9-3 Port 3 Pin Functions**

### 9.4.2 Register Configuration

Table 9-6 shows the port 3 register configuration.

**Table 9-6 Port 3 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 3 data direction register | P3DDR | W | H'C0 | H'FEB2 |
| Port 3 data register | P3DR | R/W | H'C0 | H'FF62 |
| Port 3 register | PORT3 | R | Undefined | H'FF52 |
| Port 3 open drain control register | P3ODR | R/W | H'C0 | H'FF76 |

Note: * Lower 16 bits of the address.

**Port 3 Data Direction Register (P3DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35DDR | P34DDR | P33DDR | P32DDR | P31DDR | P30DDR |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | W | W | W | W | W | W |

P3DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 3. Bits 7 and 6 are reserved. P3DDR cannot be read; if it is, an undefined value will be read.

Setting a P3DDR bit to 1 makes the corresponding port 3 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P3DDR is initialized to H'C0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the SCI is initialized, the pin states are determined by the P3DDR and P3DR specifications.

**Port 3 Data Register (P3DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

P3DR is an 8-bit readable/writable register that stores output data for the port 3 pins ($P3_5$ to $P3_0$).

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

P3DR is initialized to H'C0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 3 Register (PORT3)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35 | P34 | P33 | P32 | P31 | P30 |
| Initial value | : | 1 | 1 | —* | —* | —* | —* | —* | —* |
| R/W | : | — | — | R | R | R | R | R | R |

Note: * Determined by state of pins $P3_5$ to $P3_0$.

PORT3 is an 8-bit read-only register that shows the pin states. Writing of output data for the port 3 pins ($P3_5$ to $P3_0$) must always be performed on P3DR.

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

If a port 3 read is performed while P3DDR bits are set to 1, the P3DR values are read. If a port 3 read is performed while P3DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT3 contents are determined by the pin states, as P3DDR and P3DR are initialized. PORT3 retains its prior state after a manual reset, and in software standby mode.

**Port 3 Open Drain Control Register (P3ODR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

P3ODR is an 8-bit readable/writable register that controls the PMOS on/off status for each port 3 pin ($P3_5$ to $P3_0$).

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

Setting a P3ODR bit to 1 makes the corresponding port 3 pin an NMOS open-drain output pin, while clearing the bit to 0 makes the pin a CMOS output pin.

P3ODR is initialized to H'C0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.4.3 Pin Functions

Port 3 pins also function as SCI I/O pins ($TxD_0$, $RxD_0$, $SCK_0$, $TxD_1$, $RxD_1$, and $SCK_1$). Port 3 pin functions are shown in table 9-7.

**Table 9-7 Port 3 Pin Functions**

**Pin** | **Selection Method and Pin Functions**

$P3_5/SCK_1$

The pin function is switched as shown below according to the combination of bit C/$\overline{A}$ in the SCI1 SMR, bits CKE0 and CKE1 in SCR, and bit P35DDR.

| CKE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ | 0 | | | 1 | — |
| CKE0 | 0 | | 1 | — | — |
| P35DDR | 0 | 1 | — | — | — |
| Pin function | $P3_5$ input pin | $P3_5$ output pin* | $SCK_1$ output pin* | $SCK_1$ output pin* | $SCK_1$ input pin |

Note: * When P35ODR = 1, the pin becomes an NMOS open-drain output.

$P3_4/SCK_0$

The pin function is switched as shown below according to the combination of bit C/$\overline{A}$ in the SCI0 SMR, bits CKE0 and CKE1 in SCR, and bit P34DDR.

| CKE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ | 0 | | | 1 | — |
| CKE0 | 0 | | 1 | — | — |
| P34DDR | 0 | 1 | — | — | — |
| Pin function | $P3_4$ input pin | $P3_4$ output pin* | $SCK_0$ output pin* | $SCK_0$ output pin* | $SCK_0$ input pin |

Note: * When P34ODR = 1, the pin becomes an NMOS open-drain output.

**Table 9-7 Port 3 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P3_3/RxD_1$ | The pin function is switched as shown below according to the combination of bit RE in the SCI1 SCR, and bit P33DDR. |

| RE | 0 | | 1 |
|---|---|---|---|
| P33DDR | 0 | 1 | — |
| Pin function | $P3_3$ input pin | $P3_3$ output pin* | $RxD_1$ input pin |

Note: * When P33ODR = 1, the pin becomes an NMOS open-drain output.

$P3_2/RxD_0$ The pin function is switched as shown below according to the combination of bit RE in the SCI0 SCR, and bit P32DDR.

| RE | 0 | | 1 |
|---|---|---|---|
| P32DDR | 0 | 1 | — |
| Pin function | $P3_2$ input pin | $P3_2$ output pin* | $RxD_0$ input pin |

Note: * When P32ODR = 1, the pin becomes an NMOS open-drain output.

$P3_1/TxD_1$ The pin function is switched as shown below according to the combination of bit TE in the SCI1 SCR, and bit P31DDR.

| TE | 0 | | 1 |
|---|---|---|---|
| P31DDR | 0 | 1 | — |
| Pin function | $P3_1$ input pin | $P3_1$ output pin* | $TxD_1$ output pin |

Note: * When P31ODR = 1, the pin becomes an NMOS open-drain output.

$P3_0/TxD_0$ The pin function is switched as shown below according to the combination of bit TE in the SCI0 SCR, and bit P30DDR.

| TE | 0 | | 1 |
|---|---|---|---|
| P30DDR | 0 | 1 | — |
| Pin function | $P3_0$ input pin | $P3_0$ output pin* | $TxD_0$ output pin |

Note: * When P30ODR = 1, the pin becomes an NMOS open-drain output.

## 9.5 Port 4

### 9.5.1 Overview

Port 4 is an 8-bit input-only port. Port 4 pins also function as A/D converter analog input pins ($AN_0$ to $AN_7$) and D/A converter analog output pins ($DA_0$ and $DA_1$). Port 4 pin functions are the same in all operating modes. Figure 9-4 shows the port 4 pin configuration.

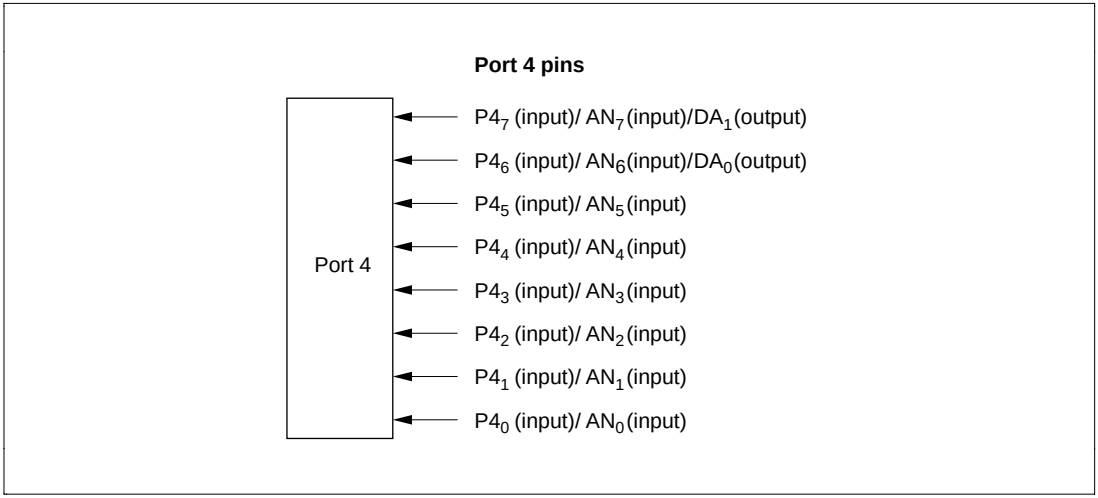



**Figure 9-4 Port 4 Pin Functions**

### 9.5.2 Register Configuration

Table 9-8 shows the port 4 register configuration. Port 4 is an input-only port, and does not have a data direction register or data register.

**Table 9-8 Port 4 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 4 register | PORT4 | R | Undefined | H'FF53 |

Note: * Lower 16 bits of the address.

**Port 4 Register (PORT4):** The pin states are always read when a port 4 read is performed.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P4_7$ to $P4_0$.

### 9.5.3 Pin Functions

Port 4 pins also function as A/D converter analog input pins ($AN_0$ to $AN_7$) and D/A converter analog output pins ($DA_0$ and $DA_1$).

## 9.6 Port 5

### 9.6.1 Overview

Port 5 is a 4-bit I/O port. Port 5 pins also function as SCI I/O pins ($TxD_2$, $RxD_2$, and $SCK_2$) and the A/D converter input pin ($\overline{ADTRG}$). Port 5 pin functions are the same in all operating modes. Figure 9-5 shows the port 5 pin configuration.

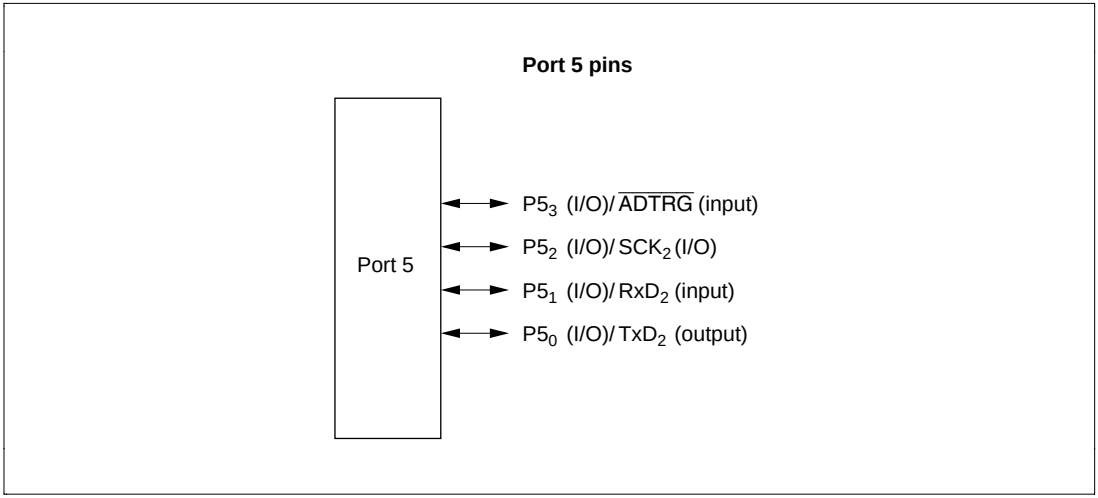



**Figure 9-5 Port 5 Pin Functions**

### 9.6.2 Register Configuration

Table 9-9 shows the port 5 register configuration.

**Table 9-9 Port 5 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 5 data direction register | P5DDR | W | H'F0 | H'FEB4 |
| Port 5 data register | P5DR | R/W | H'F0 | H'FF64 |
| Port 5 register | PORT5 | R | Undefined | H'FF54 |

Note: * Lower 16 bits of the address.

**Port 5 Data Direction Register (P5DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | P53DDR | P52DDR | P51DDR | P50DDR |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | W | W | W | W |

P5DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 5. Bits 7 to 4 are reserved. P5DDR cannot be read; if it is, an undefined value will be read.

Setting a P5DDR bit to 1 makes the corresponding port 5 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P5DDR is initialized to H'F0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the SCI is initialized, the pin states are determined by the P5DDR and P5DR specifications.

**Port 5 Data Register (P5DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | P53DR | P52DR | P51DR | P50DR |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

P5DR is an 8-bit readable/writable register that stores output data for the port 5 pins ($P5_3$ to $P5_0$).

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

P5DR is initialized to H'F0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 5 Register (PORT5)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | P53 | P52 | P51 | P50 |
| Initial value | : | 1 | 1 | 1 | 1 | —* | —* | —* | —* |
| R/W | : | — | — | — | — | R | R | R | R |

Note: * Determined by state of pins $P5_3$ to $P5_0$.

PORT5 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 5 pins ($P5_3$ to $P5_0$) must always be performed on P5DR.

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

If a port 5 read is performed while P5DDR bits are set to 1, the P5DR values are read. If a port 5 read is performed while P5DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT5 contents are determined by the pin states, as P5DDR and P5DR are initialized. PORT5 retains its prior state after a manual reset, and in software standby mode.

### 9.6.3 Pin Functions

Port 5 pins also function as SCI I/O pins ($TxD_2$, $RxD_2$, and $SCK_2$), and the A/D converter input pin ($\overline{ADTRG}$). Port 5 pin functions are shown in table 9-10.

**Table 9-10 Port 5 Pin Functions**

**Pin** | **Selection Method and Pin Functions**

$P5_3/\overline{ADTRG}$

The pin function is switched as shown below according to the combination of bits TRGS1 and TRGS0 in the A/D converter ADCR, and bit P53DDR.

| P53DDR | 0 | 1 |
|---|---|---|
| Pin function | $P5_3$ input pin | $P5_3$ output pin |
| | $\overline{ADTRG}$ input pin* | |

Note: * $\overline{ADTRG}$ input when TRGS0 = TRGS1 = 1.

$P5_2/SCK_2$

The pin function is switched as shown below according to the combination of bit C/$\overline{A}$ in the SCI2 SMR, bits CKE0 and CKE1 in SCR, and bit P52DDR.

| CKE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ | 0 | | | 1 | — |
| CKE0 | 0 | | 1 | — | — |
| P52DDR | 0 | 1 | — | — | — |
| Pin function | $P5_2$ input pin | $P5_2$ output pin | $SCK_2$ output pin | $SCK_2$ output pin | $SCK_2$ input pin |

$P5_1/RxD_2$

The pin function is switched as shown below according to the combination of bit RE in the SCI2 SCR, and bit P51DDR.

| RE | 0 | | 1 |
|---|---|---|---|
| P51DDR | 0 | 1 | — |
| Pin function | $P5_1$ input pin | $P5_1$ output pin | $RxD_2$ input pin |

$P5_0/TxD_2$

The pin function is switched as shown below according to the combination of bit TE in the SCI2 SCR, and bit P50DDR.

| TE | 0 | | 1 |
|---|---|---|---|
| P50DDR | 0 | 1 | — |
| Pin function | $P5_0$ input pin | $P5_0$ output pin | $TxD_2$ output pin |

## 9.7 Port 6

### 9.7.1 Overview

Port 6 is an 8-bit I/O port. Port 6 pins also function as interrupt input pins ($\overline{IRQ_0}$ to $\overline{IRQ_3}$), DMAC I/O pins ($\overline{DREQ_0}$, $\overline{TEND_0}$, $\overline{DREQ_1}$, and $\overline{TEND_1}$), and bus control output pins ($\overline{CS_4}$ to $\overline{CS_7}$). The functions of pins $P6_5$ to $P6_2$ are the same in all operating modes, while the functions of pins $P6_7$, $P6_6$, $P6_1$, and $P6_0$ change according to the operating mode. Pins $P6_7$ to $P6_4$ are schmitt-triggered inputs. Figure 9-6 shows the port 6 pin configuration.

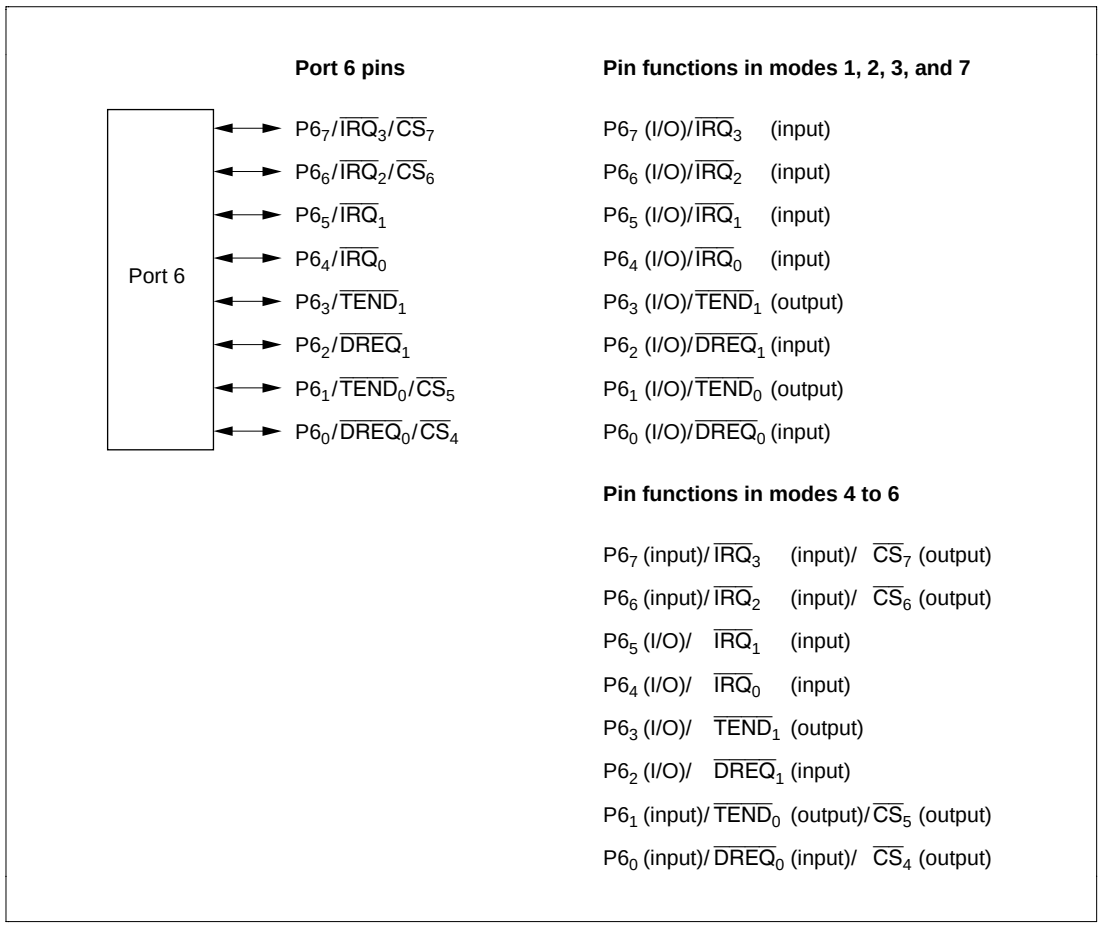



**Figure 9-6 Port 6 Pin Functions**

### 9.7.2 Register Configuration

Table 9-11 shows the port 6 register configuration.

**Table 9-11 Port 6 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 6 data direction register | P6DDR | W | H'00 | H'FEB5 |
| Port 6 data register | P6DR | R/W | H'00 | H'FF65 |
| Port 6 register | PORT6 | R | Undefined | H'FF55 |

Note: * Lower 16 bits of the address.

**Port 6 Data Direction Register (P6DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P67DDR | P66DDR | P65DDR | P64DDR | P63DDR | P62DDR | P61DDR | P60DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P6DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 6. P6DDR cannot be read; if it is, an undefined value will be read.

Setting a P6DDR bit to 1 makes the corresponding port 6 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P6DDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the DMAC is initialized by a manual reset, the pin states are determined by the P6DDR and P6DR specifications.

**Port 6 Data Register (P6DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P67DR | P66DR | P65DR | P64DR | P63DR | P62DR | P61DR | P60DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

P6DR is an 8-bit readable/writable register that stores output data for the port 6 pins ($P6_7$ to $P6_0$).

P6DR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 6 Register (PORT6)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| RW | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P6_7$ to $P6_0$.

PORT6 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 6 pins ($P6_7$ to $P6_0$) must always be performed on P6DR.

If a port 6 read is performed while P6DDR bits are set to 1, the P6DR values are read. If a port 6 read is performed while P6DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT6 contents are determined by the pin states, as P6DDR and P6DR are initialized. PORT6 retains its prior state after a manual reset, and in software standby mode.

### 9.7.3 Pin Functions

Port 6 pins also function as interrupt input pins ($\overline{IRQ_0}$ to $\overline{IRQ_3}$), DMAC I/O pins ($\overline{DREQ_0}$, $\overline{TEND_0}$, $\overline{DREQ_1}$, and $\overline{TEND_1}$), and bus control output pins ($\overline{CS_4}$ to $\overline{CS_7}$). Port 6 pin functions are shown in table 9-12.

**Table 9-12 Port 6 Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P6_7/\overline{IRQ_3}/\overline{CS_7}$ | The pin function is switched as shown below according to bit P67DDR. |

| Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| P67DDR | 0 | 1 | 0 | 1 |
| Pin function | $P6_7$ input pin | $P6_7$ output pin | $P6_7$ input pin | $\overline{CS_7}$ output pin |
| | $\overline{IRQ_3}$ interrupt input pin | | | |

| $P6_6/\overline{IRQ_2}/\overline{CS_6}$ | The pin function is switched as shown below according to bit P66DDR. |
|---|---|

| Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| P66DDR | 0 | 1 | 0 | 1 |
| Pin function | $P6_6$ input pin | $P6_6$ output pin | $P6_6$ input pin | $\overline{CS_6}$ output pin |
| | $\overline{IRQ_2}$ interrupt input pin | | | |

| $P6_5/\overline{IRQ_1}$ | The pin function is switched as shown below according to bit P65DDR. |
|---|---|

| P65DDR | 0 | 1 |
|---|---|---|
| Pin function | $P6_5$ input pin | $P6_5$ output pin |
| | $\overline{IRQ_1}$ interrupt input pin | |

| $P6_4/\overline{IRQ_0}$ | The pin function is switched as shown below according to bit P64DDR. |
|---|---|

| P64DDR | 0 | 1 |
|---|---|---|
| Pin function | $P6_4$ input pin | $P6_4$ output pin |
| | $\overline{IRQ_0}$ interrupt input pin | |

**Table 9-12 Port 6 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P6_3/\overline{TEND}_1$ | The pin function is switched as shown below according to the combination of bit TEE1 in the DMAC DMATCR, and bit P63DDR. |

| TEE1 | 0 | | 1 |
|---|---|---|---|
| P63DDR | 0 | 1 | — |
| Pin function | $P6_3$ input pin | $P6_3$ output pin | $\overline{TEND}_1$ output |

| $P6_2/\overline{DREQ}_1$ | The pin function is switched as shown below according to bit P62DDR. |
|---|---|

| P62DDR | 0 | 1 |
|---|---|---|
| Pin function | $P6_2$ input pin | $P6_2$ output pin |
| | $\overline{DERQ}_1$ input | |

| $P6_1/\overline{TEND}_0/\overline{CS}_5$ | The pin function is switched as shown below according to the combination of bit TEE0 in the DMAC DMATCR, and bit P61DDR. |
|---|---|

Modes 1, 2, 3, 7

| TEE0 | 0 | | 1 |
|---|---|---|---|
| P61DDR | 0 | 1 | — |
| Pin function | $P6_1$ input pin | $P6_1$ output pin | $\overline{TEND}_0$ output |

Modes 4 to 6

| TEE0 | 0 | | 1 |
|---|---|---|---|
| P61DDR | 0 | 1 | — |
| Pin function | $P6_1$ input pin | $\overline{CS}_5$ output pin | $\overline{TEND}_0$ output |

| $P6_0/\overline{DREQ}_0/\overline{CS}_4$ | The pin function is switched as shown below according to bit P60DDR. |
|---|---|

| Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| P60DDR | 0 | 1 | 0 | 1 |
| Pin function | $P6_0$ input pin | $P6_0$ output pin | $P6_0$ input pin | $\overline{CS}_4$ output pin |
| | $\overline{DREQ}_0$ input | | | |

## 9.8 Port A

### 9.8.1 Overview

Port A is an 8-bit I/O port. Port A pins also function as address bus outputs and interrupt input pins ($\overline{IRQ_4}$ to $\overline{IRQ_7}$). The pin functions change according to the operating mode.

Port A has a built-in MOS input pull-up function that can be controlled by software. Pins $PA_7$ to $PA_4$ are schmitt-triggered inputs.

Figure 9-7 shows the port A pin configuration.

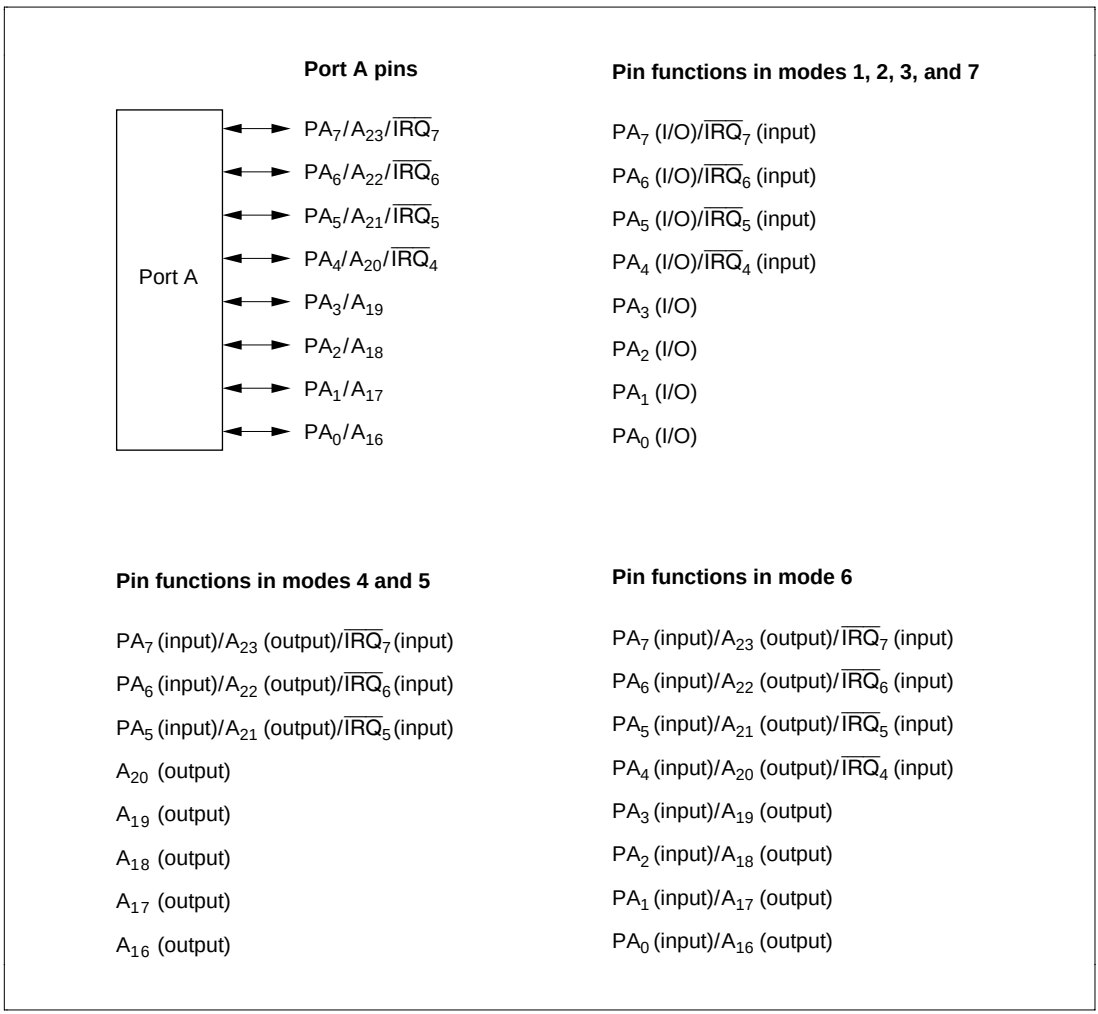



**Figure 9-7 Port A Pin Functions**

### 9.8.2 Register Configuration

Table 9-13 shows the port A register configuration.

**Table 9-13 Port A Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port A data direction register | PADDR | W | H'00 | H'FEB9 |
| Port A data register | PADR | R/W | H'00 | H'FF69 |
| Port A register | PORTA | R | Undefined | H'FF59 |
| Port A MOS pull-up control register | PAPCR | R/W | H'00 | H'FF70 |
| Port A open-drain control register | PAODR | R/W | H'00 | H'FF77 |

Note: * Lower 16 bits of the address.

**Port A Data Direction Register (PADDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PA7DDR | PA6DDR | PA5DDR | PA4DDR | PA3DDR | PA2DDR | PA1DDR | PA0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PADDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port A. PADDR cannot be read; if it is, an undefined value will be read.

PADDR is initialized to H'00 by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 2, 3, and 7

  Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 4 and 5

  The corresponding port A pins are address outputs irrespective of the value of bits PA4DDR to PA0DDR.

  Setting one of bits PA7DDR to PA5DDR to 1 makes the corresponding port A pin an address output, while clearing the bit to 0 makes the pin an input port.

- Mode 6

  Setting a PADDR bit to 1 makes the corresponding port A pin an address output while clearing the bit to 0 makes the pin an input port.

**Port A Data Register (PADR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PADR is an 8-bit readable/writable register that stores output data for the port A pins ($PA_7$ to $PA_0$).

PADR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port A Register (PORTA)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PA_7$ to $PA_0$.

PORTA is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port A pins ($PA_7$ to $PA_0$) must always be performed on PADR.

If a port A read is performed while PADDR bits are set to 1, the PADR values are read. If a port A read is performed while PADDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTA contents are determined by the pin states, as PADDR and PADR are initialized. PORTA retains its prior state after a manual reset, and in software standby mode.

**Port A MOS Pull-Up Control Register (PAPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PA7PCR | PA6PCR | PA5PCR | PA4PCR | PA3PCR | PA2PCR | PA1PCR | PA0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PAPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port A on an individual bit basis.

All the bits are valid in modes 1, 2, 3, 6, and 7, and bits 7 to 5 are valid in modes 4 and 5. When a PADDR bit is cleared to 0 (input port setting), setting the corresponding PAPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PAPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port A Open Drain Control Register (PAODR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PA7ODR | PA6ODR | PA5ODR | PA4ODR | PA3ODR | PA2ODR | PA1ODR | PA0ODR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PAODR is an 8-bit readable/writable register that controls whether PMOS is on or off for each port A pin ($PA_7$ to $PA_0$).

All bits are valid in modes 1, 2, 3, and 7.

Setting a PAODR bit to 1 makes the corresponding port A pin an NMOS open-drain output, while clearing the bit to 0 makes the pin a CMOS output.

PAODR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.8.3 Pin Functions

**Modes 1, 2, 3 and 7:** In mode 1, 2, 3, and 7, port A pins function as I/O ports and interrupt input pins. Input or output can be specified for each pin on an individual bit basis. Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.

Port A pin functions in modes 1, 2, 3, and 7 are shown in figure 9-8.

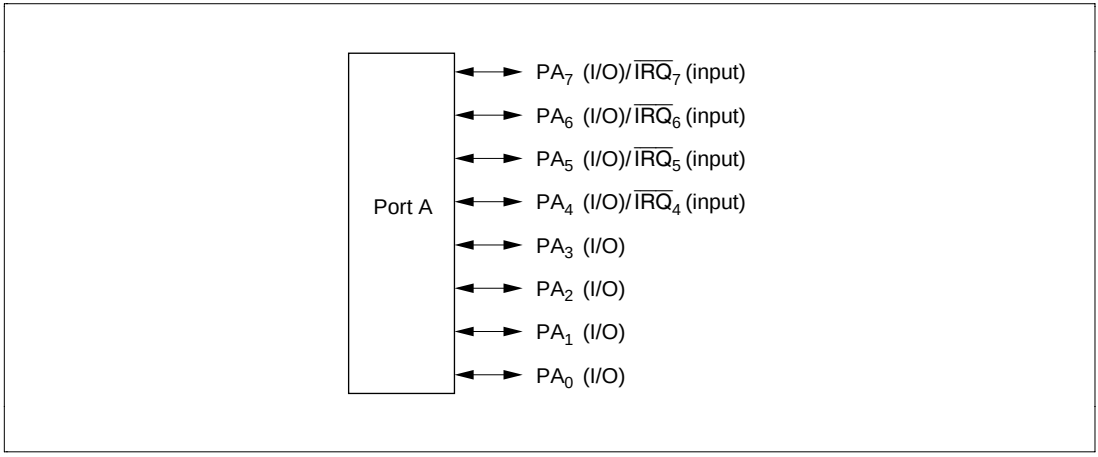



**Figure 9-8 Port A Pin Functions (Modes 1, 2, 3, and 7)**

**Modes 4 and 5:** In modes 4 and 5, the lower 5 bits of port A are designated as address outputs automatically, while the upper 3 bits function as address outputs or input ports and interrupt input pins. Input or output can be specified individually for the upper 3 bits. Setting one of bits PA7DDR to PA5DDR to 1 makes the corresponding port A pin an address output, while clearing the bit to 0 makes the pin an input port.

Port A pin functions in modes 4 and 5 are shown in figure 9-9.

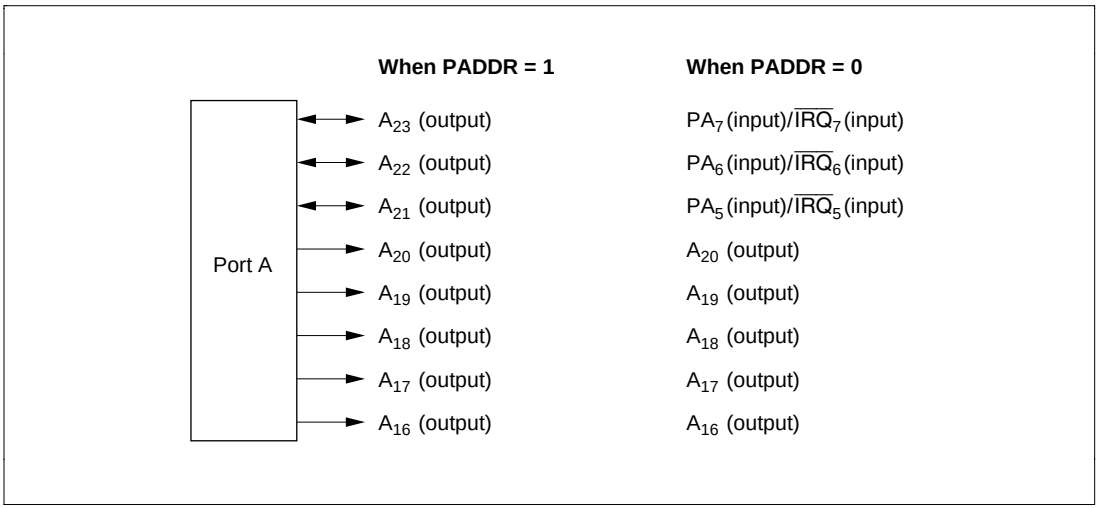



**Figure 9-9 Port A Pin Functions (Modes 4 and 5)**

**Mode 6:** In mode 6, port A pins function as address outputs or input ports and interrupt input pins. Input or output can be specified on an individual bit basis. Setting a PADDR bit to 1 makes the corresponding port A pin an address output, while clearing the bit to 0 makes the pin an input port.

Port A pin functions in mode 6 are shown in figure 9-10.

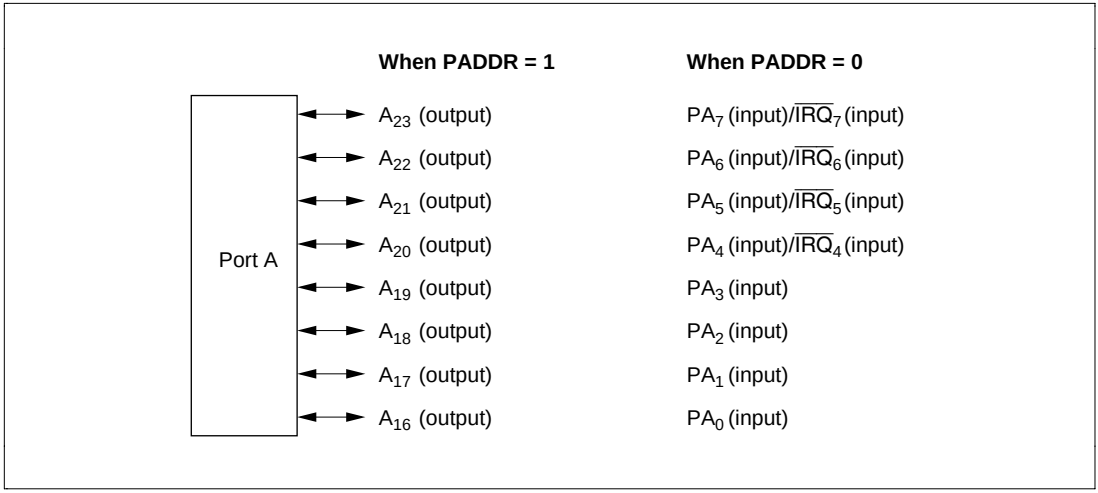



**Figure 9-10 Port A Pin Functions (Mode 6)**

### 9.8.4 MOS Input Pull-Up Function

Port A has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used by pins $PA_7$ to $PA_5$ in modes 4 and 5, and by all pins in modes 1, 2, 3, 6, and 7. MOS input pull-up can be specified as on or off on an individual bit basis.

When a PADDR bit is cleared to 0, setting the corresponding PAPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 9-14 summarizes the MOS input pull-up states.

**Table 9-14 MOS Input Pull-Up States (Port A)**

| Modes | | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|---|
| 1 to 3, 6, 7 | $PA_7$ to $PA_0$ | OFF | | ON/OFF | | |
| 4, 5 | $PA_7$ to $PA_5$ | | | ON/OFF | | |
| | $PA_4$ to $PA_0$ | | | OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PADDR = 0 and PAPCR = 1; otherwise off.

## 9.9 Port B

### 9.9.1 Overview

Port B is an 8-bit I/O port. Port B has an address bus output function, and the pin functions change according to the operating mode.

Port B has a built-in MOS input pull-up function that can be controlled by software.

Figure 9-11 shows the port B pin configuration.

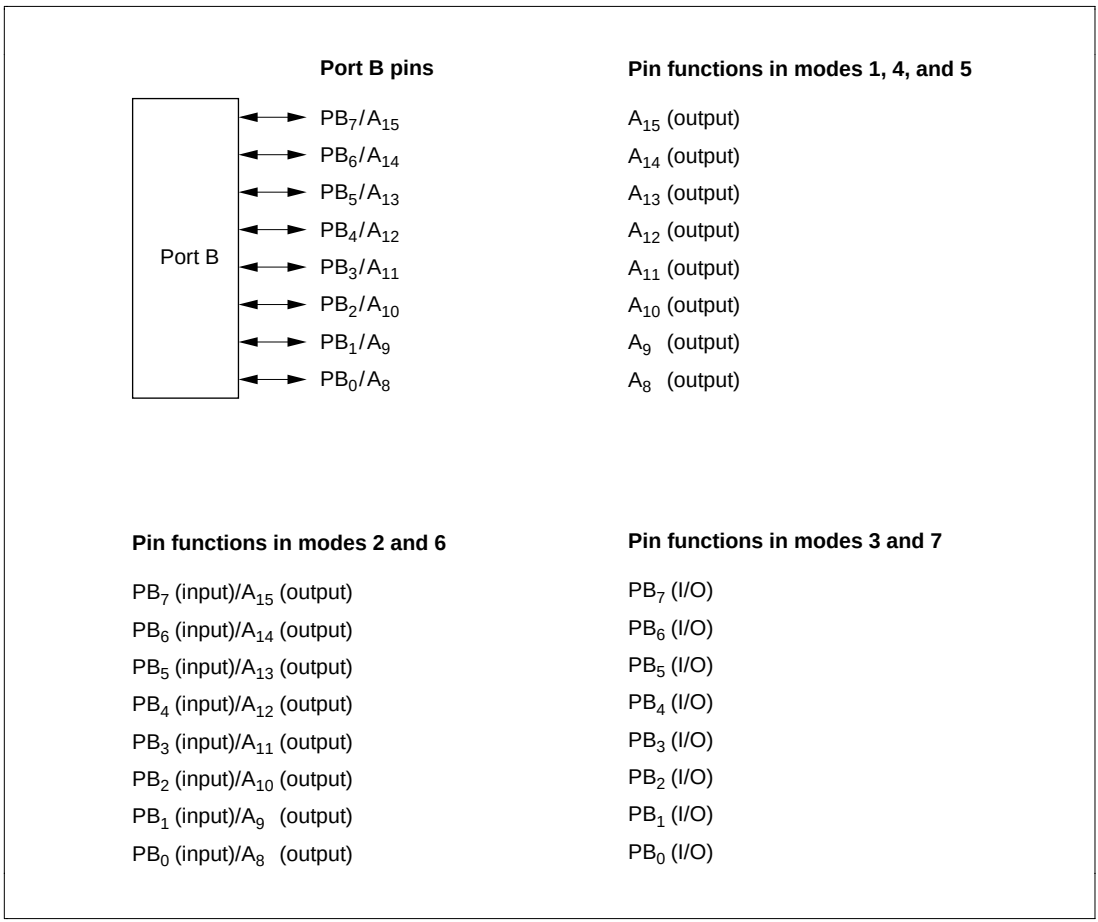



**Figure 9-11 Port B Pin Functions**

### 9.9.2 Register Configuration

Table 9-15 shows the port B register configuration.

**Table 9-15 Port B Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port B data direction register | PBDDR | W | H'00 | H'FEBA |
| Port B data register | PBDR | R/W | H'00 | H'FF6A |
| Port B register | PORTB | R | Undefined | H'FF5A |
| Port B MOS pull-up control register | PBPCR | R/W | H'00 | H'FF71 |

Note: * Lower 16 bits of the address.

**Port B Data Direction Register (PBDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PBDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port B. PBDDR cannot be read; if it is, an undefined value will be read.

PBDDR is initialized to H'00 by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 4, and 5
  The corresponding port B pins are address outputs irrespective of the value of the PBDDR bits.
- Modes 2 and 6
  Setting a PBDDR bit to 1 makes the corresponding port B pin an address output, while clearing the bit to 0 makes the pin an input port.
- Modes 3 and 7
  Setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port.

**Port B Data Register (PBDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PBDR is an 8-bit readable/writable register that stores output data for the port B pins ($PB_7$ to $PB_0$). PBDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port B Register (PORTB)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PB_7$ to $PB_0$.

PORTB is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port B pins ($PB_7$ to $PB_0$) must always be performed on PBDR.

If a port B read is performed while PBDDR bits are set to 1, the PBDR values are read. If a port B read is performed while PBDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTB contents are determined by the pin states, as PBDDR and PBDR are initialized. PORTB retains its prior state after a manual reset, and in software standby mode.

**Port B MOS Pull-Up Control Register (PBPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PBPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port B on an individual bit basis.

When a PBDDR bit is cleared to 0 (input port setting) in mode 2, 3, 6, or 7, setting the corresponding PBPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PBPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.9.3 Pin Functions

**Modes 1, 4, and 5:** In modes 1, 4, and 5, port B pins are automatically designated as address outputs.

Port B pin functions in modes 1, 4, and 5 are shown in figure 9-12.

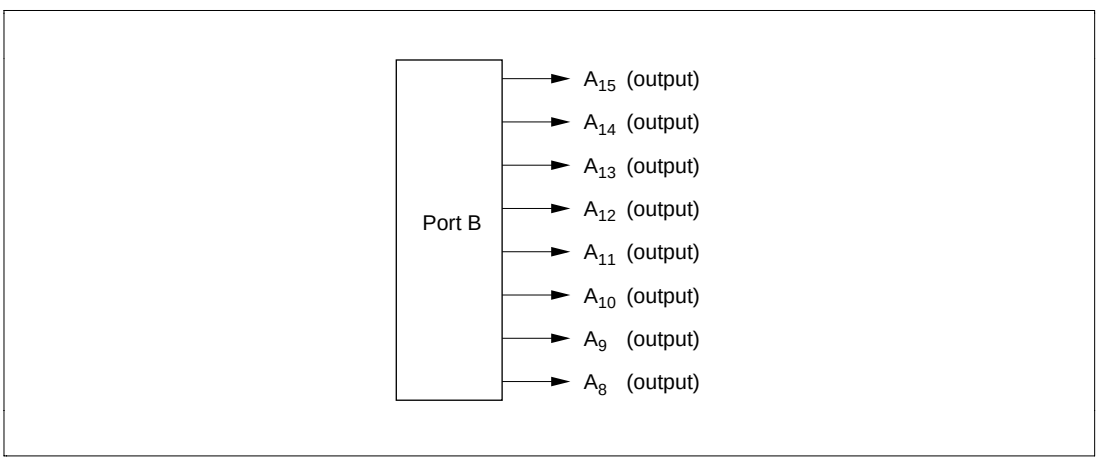



**Figure 9-12 Port B Pin Functions (Modes 1, 4, and 5)**

**Modes 2 and 6:** In modes 2 and 6, port B pins function as address outputs or input ports. Input or output can be specified on an individual bit basis. Setting a PBDDR bit to 1 makes the corresponding port B pin an address output, while clearing the bit to 0 makes the pin an input port.

Port B pin functions in modes 2 and 6 are shown in figure 9-13.

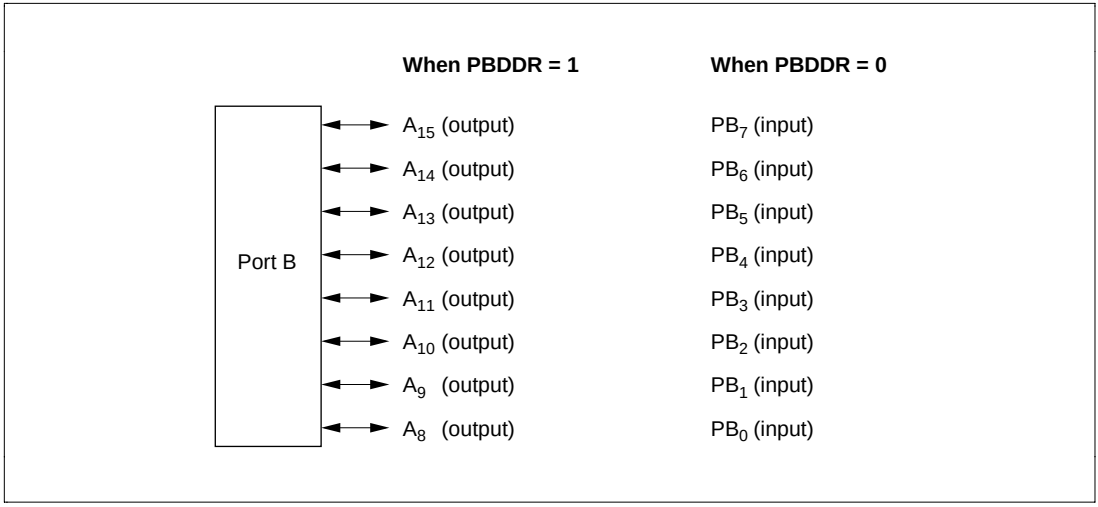



**Figure 9-13 Port B Pin Functions (Modes 2 and 6)**

**Modes 3 and 7:** In modes 3 and 7, port B pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port.

Port B pin functions in modes 3 and 7 are shown in figure 9-14.

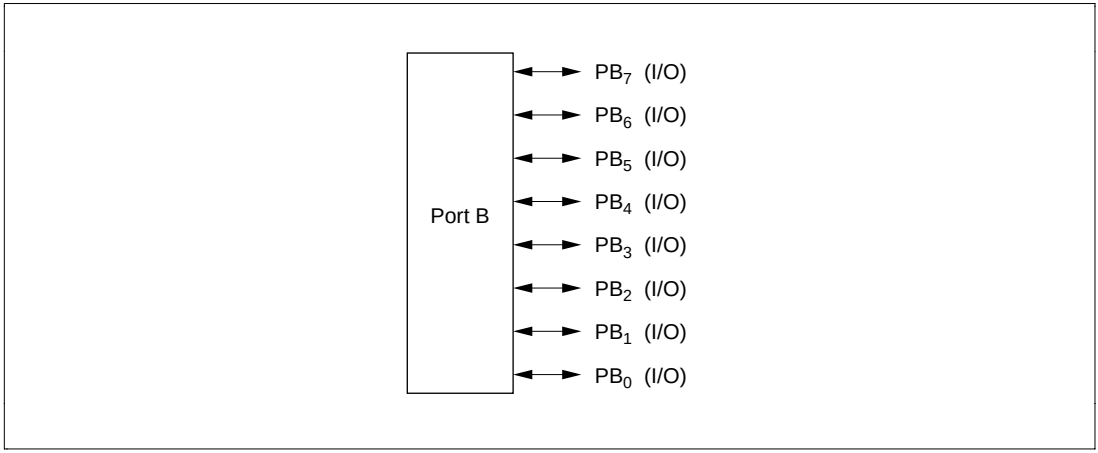



**Figure 9-14 Port B Pin Functions (Modes 3 and 7)**

### 9.9.4 MOS Input Pull-Up Function

Port B has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 2, 3, 6, and 7, and can be specified as on or off on an individual bit basis.

When a PBDDR bit is cleared to 0 in mode 2, 3, 6, or 7, setting the corresponding PBPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 9-16 summarizes the MOS input pull-up states.

**Table 9-16 MOS Input Pull-Up States (Port B)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 4, 5 | OFF | | OFF | | |
| 2, 3, 6, 7 | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PBDDR = 0 and PBPCR = 1; otherwise off.

## 9.10 Port C

### 9.10.1 Overview

Port C is an 8-bit I/O port. Port C has an address bus output function, and the pin functions change according to the operating mode.

Port C has a built-in MOS input pull-up function that can be controlled by software.

Figure 9-15 shows the port C pin configuration.

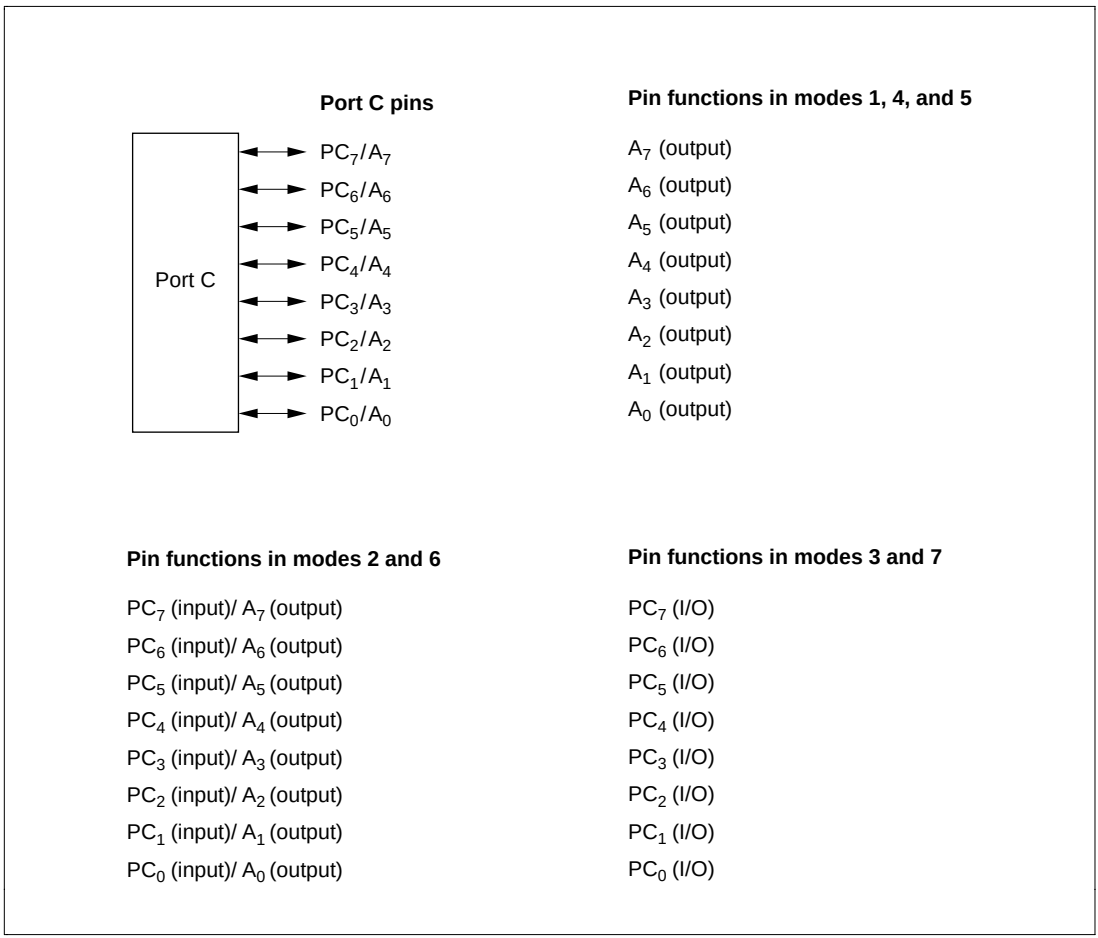



**Figure 9-15 Port C Pin Functions**

### 9.10.2 Register Configuration

Table 9-17 shows the port C register configuration.

**Table 9-17 Port C Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port C data direction register | PCDDR | W | H'00 | H'FEBB |
| Port C data register | PCDR | R/W | H'00 | H'FF6B |
| Port C register | PORTC | R | Undefined | H'FF5B |
| Port C MOS pull-up control register | PCPCR | R/W | H'00 | H'FF72 |

Note: * Lower 16 bits of the address.

**Port C Data Direction Register (PCDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PCDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port C. PCDDR cannot be read; if it is, an undefined value will be read.

PCDDR is initialized to H'00 by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 4, and 5
  The corresponding port C pins are address outputs irrespective of the value of the PCDDR bits.
- Modes 2 and 6
  Setting a PCDDR bit to 1 makes the corresponding port C pin an address output, while clearing the bit to 0 makes the pin an input port.
- Modes 3 and 7
  Setting a PCDDR bit to 1 makes the corresponding port C pin an output port, while clearing the bit to 0 makes the pin an input port.

**Port C Data Register (PCDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PCDR is an 8-bit readable/writable register that stores output data for the port C pins ($PC_7$ to $PC_0$).

PCDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port C Register (PORTC)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PC_7$ to $PC_0$.

PORTC is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port C pins ($PC_7$ to $PC_0$) must always be performed on PCDR.

If a port C read is performed while PCDDR bits are set to 1, the PCDR values are read. If a port C read is performed while PCDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTC contents are determined by the pin states, as PCDDR and PCDR are initialized. PORTC retains its prior state after a manual reset, and in software standby mode.

**Port C MOS Pull-Up Control Register (PCPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PCPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port C on an individual bit basis.

When a PCDDR bit is cleared to 0 (input port setting) in mode 2, 3, 6, or 7, setting the corresponding PCPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PCPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.10.3 Pin Functions

**Modes 1, 4, and 5:** In modes 1, 4, and 5, port C pins are automatically designated as address outputs.

Port C pin functions in modes 1, 4, and 5 are shown in figure 9-16.

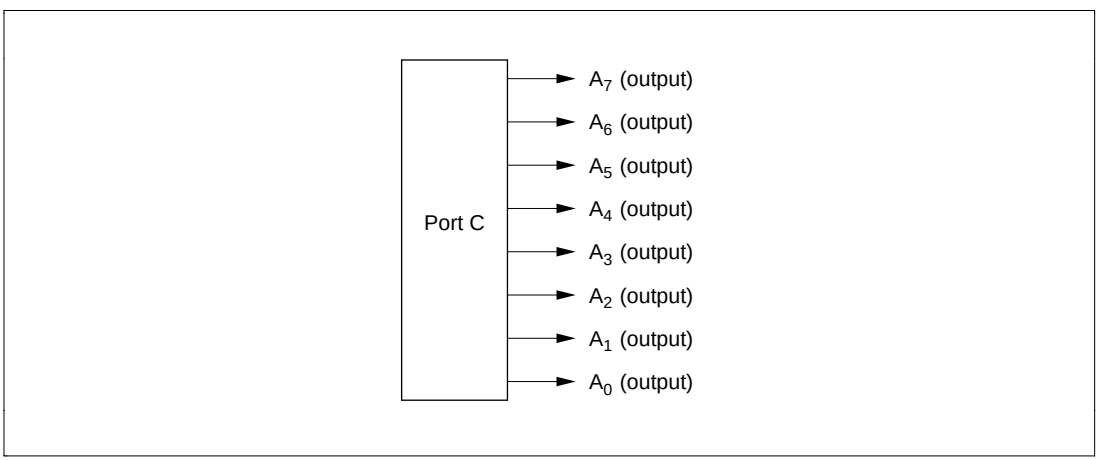



**Figure 9-16 Port C Pin Functions (Modes 1, 4, and 5)**

**Modes 2 and 6:** In modes 2 and 6, port C pins function as address outputs or input ports. Input or output can be specified on an individual bit basis. Setting a PCDDR bit to 1 makes the corresponding port C pin an address output, while clearing the bit to 0 makes the pin an input port.

Port C pin functions in modes 2 and 6 are shown in figure 9-17.

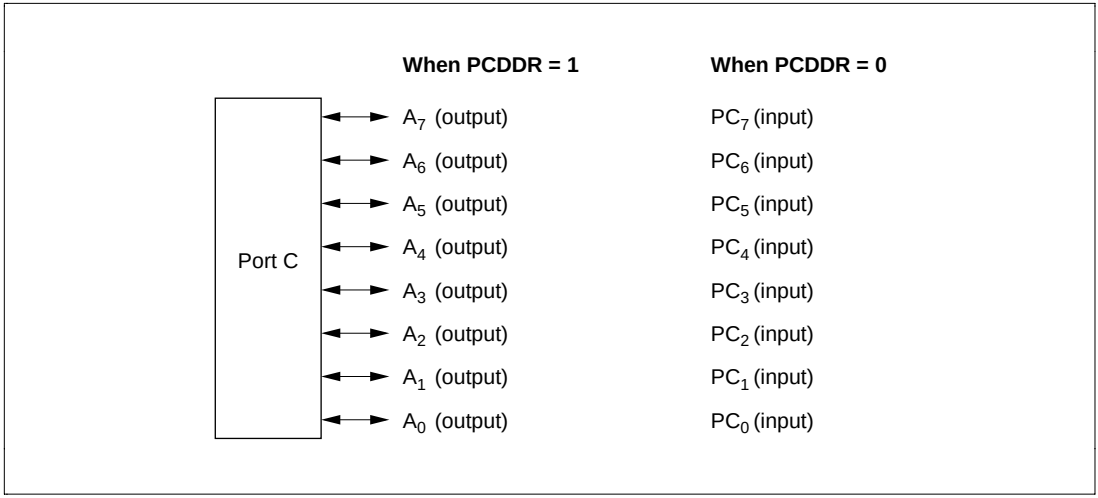



**Figure 9-17 Port C Pin Functions (Modes 2 and 6)**

**Modes 3 and 7:** In modes 3 and 7, port C pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PCDDR bit to 1 makes the corresponding port C pin an output port, while clearing the bit to 0 makes the pin an input port.

Port C pin functions in modes 3 and 7 are shown in figure 9-18.

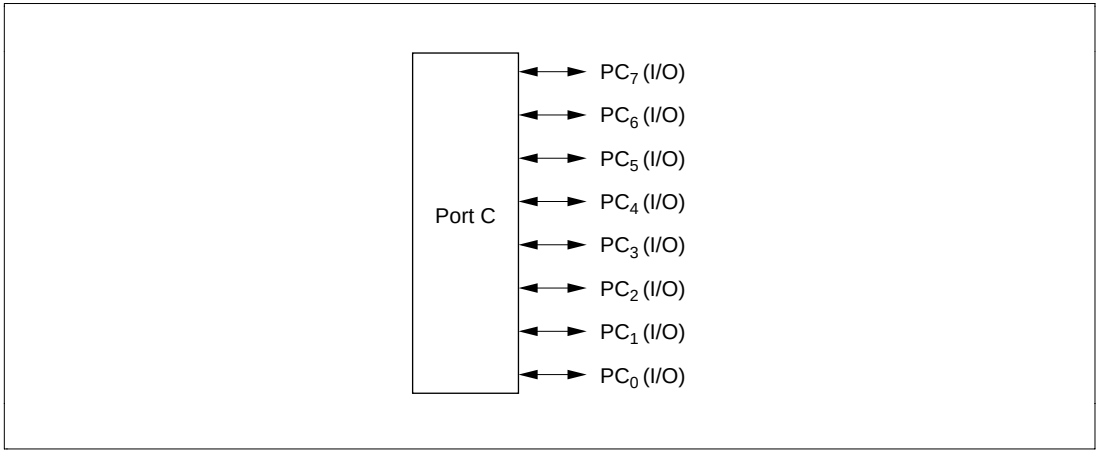



**Figure 9-18 Port C Pin Functions (Modes 3 and 7)**

### 9.10.4 MOS Input Pull-Up Function

Port C has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 2, 3, 6, and 7, and can be specified as on or off on an individual bit basis.

When a PCDDR bit is cleared to 0 in mode 2, 3, 6, or 7, setting the corresponding PCPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 9-18 summarizes the MOS input pull-up states.

**Table 9-18 MOS Input Pull-Up States (Port C)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 4, 5 | OFF | | OFF | | |
| 2, 3, 6, 7 | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PCDDR = 0 and PCPCR = 1; otherwise off.

## 9.11 Port D

### 9.11.1 Overview

Port D is an 8-bit I/O port. Port D has a data bus I/O function, and the pin functions change according to the operating mode.

Port D has a built-in MOS input pull-up function that can be controlled by software.

Figure 9-19 shows the port D pin configuration.

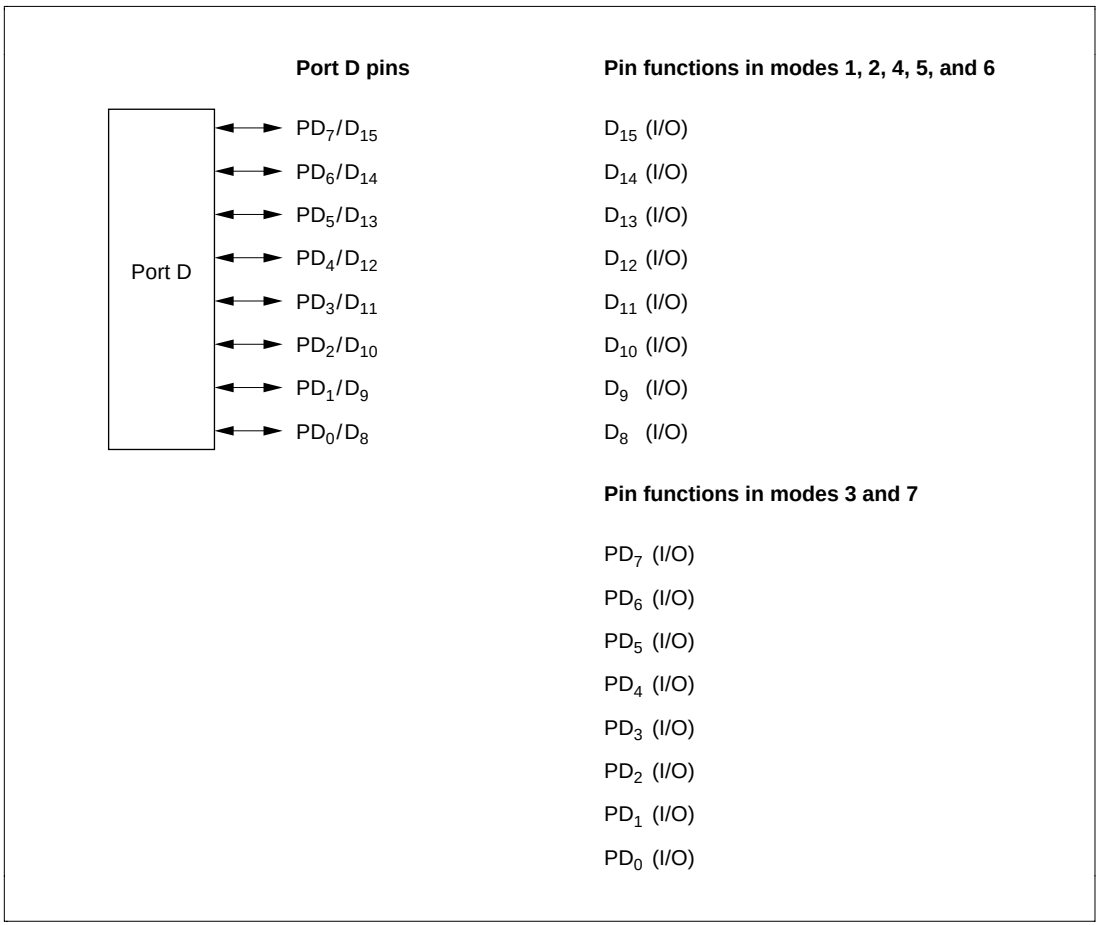



**Figure 9-19 Port D Pin Functions**

### 9.11.2 Register Configuration

Table 9-19 shows the port D register configuration.

**Table 9-19 Port D Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port D data direction register | PDDDR | W | H'00 | H'FEBC |
| Port D data register | PDDR | R/W | H'00 | H'FF6C |
| Port D register | PORTD | R | Undefined | H'FF5C |
| Port D MOS pull-up control register | PDPCR | R/W | H'00 | H'FF73 |

Note: * Lower 16 bits of the address.

**Port D Data Direction Register (PDDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PDDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port D. PDDDR cannot be read; if it is, an undefined value will be read..

PDDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

- Modes 1, 2, 4, 5, and 6

  The input/output direction specification by PDDDR is ignored, and port D is automatically designated for data I/O.

- Modes 3 and 7

  Setting a PDDDR bit to 1 makes the corresponding port D pin an output port, while clearing the bit to 0 makes the pin an input port.

**Port D Data Register (PDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PDDR is an 8-bit readable/writable register that stores output data for the port D pins ($PD_7$ to $PD_0$).

PDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port D Register (PORTD)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PD_7$ to $PD_0$.

PORTD is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port D pins ($PD_7$ to $PD_0$) must always be performed on PDDR.

If a port D read is performed while PDDDR bits are set to 1, the PDDR values are read. If a port D read is performed while PDDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTD contents are determined by the pin states, as PDDDR and PDDR are initialized. PORTD retains its prior state after a manual reset, and in software standby mode.

**Port D MOS Pull-Up Control Register (PDPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PDPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port D on an individual bit basis.

When a PDDDR bit is cleared to 0 (input port setting) in mode 3 or 7, setting the corresponding PDPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PDPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.11.3 Pin Functions

**Modes 1, 2, 4, 5, and 6:** In modes 1, 2, 4, 5, and 6, port D pins are automatically designated as data I/O pins.

Port D pin functions in modes 1, 2, 4, 5, and 6 are shown in figure 9-20.

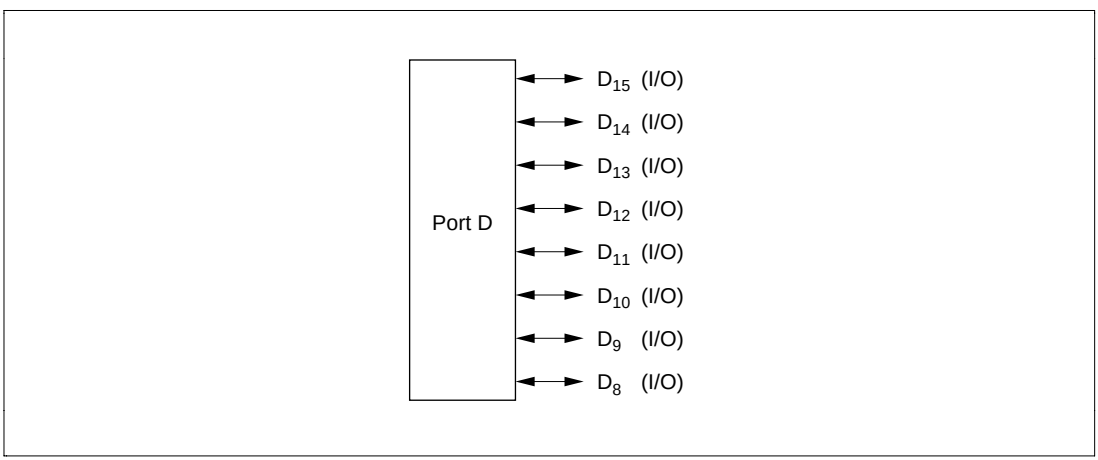



**Figure 9-20 Port D Pin Functions (Modes 1, 2, 4, 5, and 6)**

**Modes 3 and 7:** In modes 3 and 7, port D pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PDDDR bit to 1 makes the corresponding port D pin an output port, while clearing the bit to 0 makes the pin an input port.

Port D pin functions in modes 3 and 7 are shown in figure 9-21.

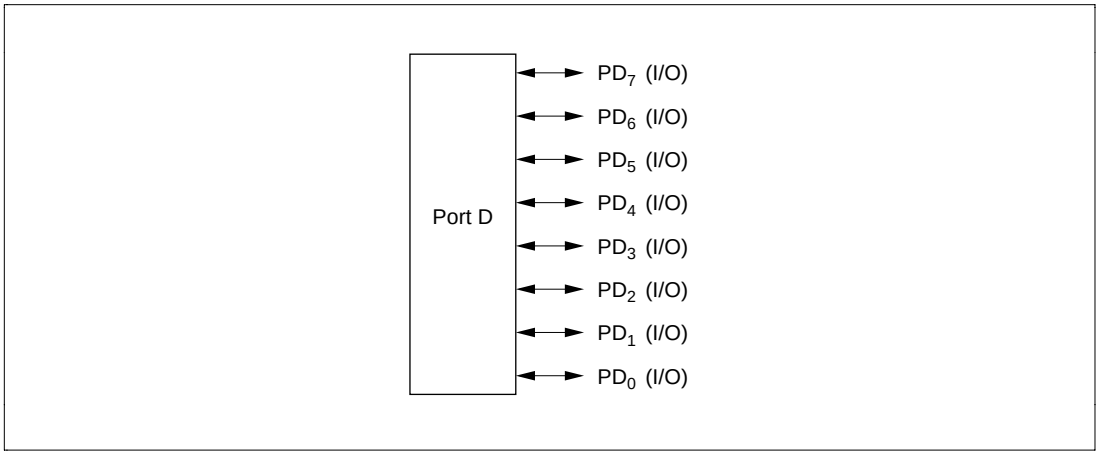



**Figure 9-21 Port D Pin Functions (Modes 3 and 7)**

### 9.11.4 MOS Input Pull-Up Function

Port D has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 3 and 7, and can be specified as on or off on an individual bit basis.

When a PDDDR bit is cleared to 0 in mode 3 or 7, setting the corresponding PDPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 9-20 summarizes the MOS input pull-up states.

**Table 9-20 MOS Input Pull-Up States (Port D)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 2, 4 to 6 | OFF | | OFF | | |
| 3, 7 | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PDDDR = 0 and PDPCR = 1; otherwise off.

## 9.12 Port E

### 9.12.1 Overview

Port E is an 8-bit I/O port. Port E has a data bus I/O function, and the pin functions change according to the operating mode and whether 8-bit or 16-bit bus mode is selected.

Port E has a built-in MOS input pull-up function that can be controlled by software.

Figure 9-22 shows the port E pin configuration.

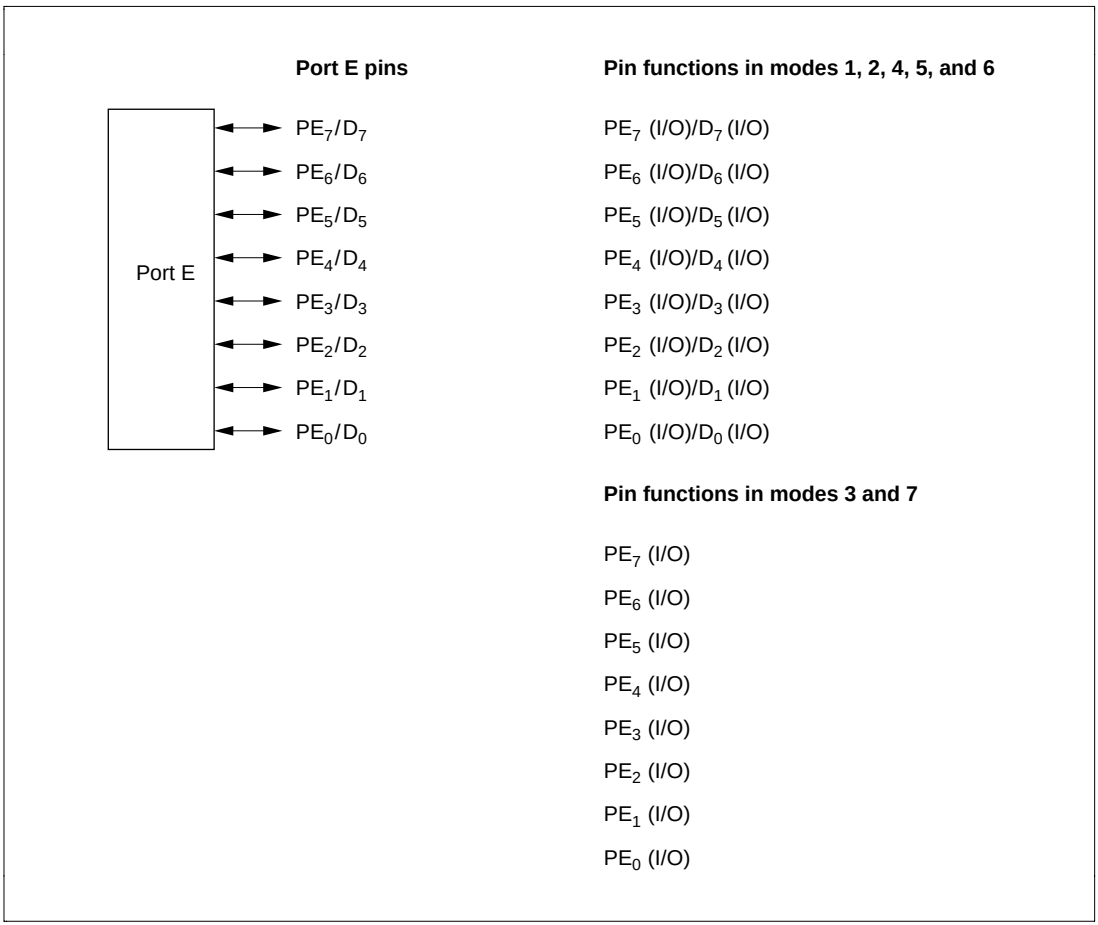



**Figure 9-22 Port E Pin Functions**

### 9.12.2 Register Configuration

Table 9-21 shows the port E register configuration.

**Table 9-21 Port E Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port E data direction register | PEDDR | W | H'00 | H'FEBD |
| Port E data register | PEDR | R/W | H'00 | H'FF6D |
| Port E register | PORTE | R | Undefined | H'FF5D |
| Port E MOS pull-up control register | PEPCR | R/W | H'00 | H'FF74 |

Note: * Lower 16 bits of the address.

**Port E Data Direction Register (PEDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PEDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port E. PEDDR cannot be read; if it is, an undefined value will be read.

PEDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

- Modes 1, 2, 4, 5, and 6

  When 8-bit bus mode has been selected, port E pins function as I/O ports. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

  When 16-bit bus mode has been selected, the input/output direction specification by PEDDR is ignored, and port E is designated for data I/O.

  For details of 8-bit and 16-bit bus modes, see section 6, Bus Controller.

- Modes 3 and 7

  Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

**Port E Data Register (PEDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PEDR is an 8-bit readable/writable register that stores output data for the port E pins ($PE_7$ to $PE_0$).

PEDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port E Register (PORTE)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PE_7$ to $PE_0$.

PORTE is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port E pins ($PE_7$ to $PE_0$) must always be performed on PEDR.

If a port E read is performed while PEDDR bits are set to 1, the PEDR values are read. If a port E read is performed while PEDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTE contents are determined by the pin states, as PEDDR and PEDR are initialized. PORTE retains its prior state after a manual reset, and in software standby mode.

**Port E MOS Pull-Up Control Register (PEPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PEPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port E on an individual bit basis.

When a PEDDR bit is cleared to 0 (input port setting) when 8-bit bus mode is selected in mode 1, 2, 4, 5, or 6, or in mode 3 or 7, setting the corresponding PEPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PEPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 9.12.3 Pin Functions

**Modes 1, 2, 4, 5, and 6:** In modes 1, 2, 4, 5, and 6, when 8-bit access is designated and 8-bit bus mode is selected, port E pins are automatically designated as I/O ports. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

When 16-bit bus mode is selected, the input/output direction specification by PEDDR is ignored, and port E is designated for data I/O.

Port E pin functions in modes 1, 2, 4, 5, and 6 are shown in figure 9-23.

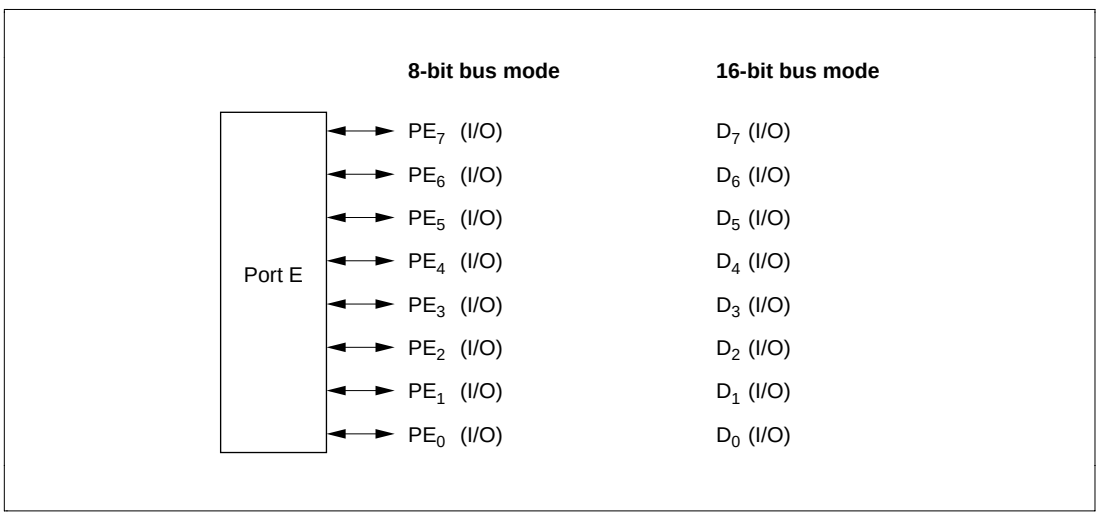



**Figure 9-23 Port E Pin Functions (Modes 1, 2, 4, 5, and 6)**

**Modes 3 and 7:** In modes 3 and 7, port E pins function as I/O ports. Input or output can be specified for each pin on a bit-by-bit basis. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

Port E pin functions in modes 3 and 7 are shown in figure 9-24.

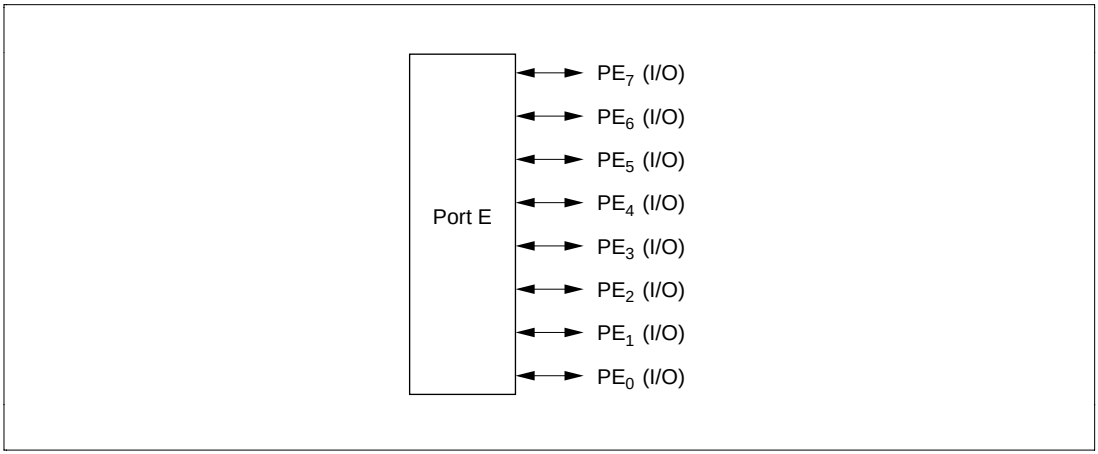



**Figure 9-24 Port E Pin Functions (Modes 3 and 7)**

### 9.12.4 MOS Input Pull-Up Function

Port E has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 1, 2, 4, 5, and 6 when 8-bit bus mode is selected, or in mode 3 or 7, and can be specified as on or off on an individual bit basis.

When a PEDDR bit is cleared to 0 in mode 1, 2, 4, 5, or 6 when 8-bit bus mode is selected, or in mode 3 or 7, setting the corresponding PEPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 9-22 summarizes the MOS input pull-up states.

**Table 9-22 MOS Input Pull-Up States (Port E)**

| Modes | | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|---|
| 3, 7 | | OFF | | ON/OFF | | |
| 1, 2, 4 to 6 | 8-bit bus | | | | | |
| | 16-bit bus | | | OFF | | |

**Legend:**
OFF : MOS input pull-up is always off.
ON/OFF : On when PEDDR = 0 and PEPCR = 1; otherwise off.

## 9.13 Port F

### 9.13.1 Overview

Port F is an 8-bit I/O port. Port F pins also function as bus control signal input/output pins ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, $\overline{LCAS}$, $\overline{WAIT}$, $\overline{BREQO}$, $\overline{BREQ}$, and $\overline{BACK}$) and the system clock (ø) output pin.

Figure 9-25 shows the port F pin configuration.

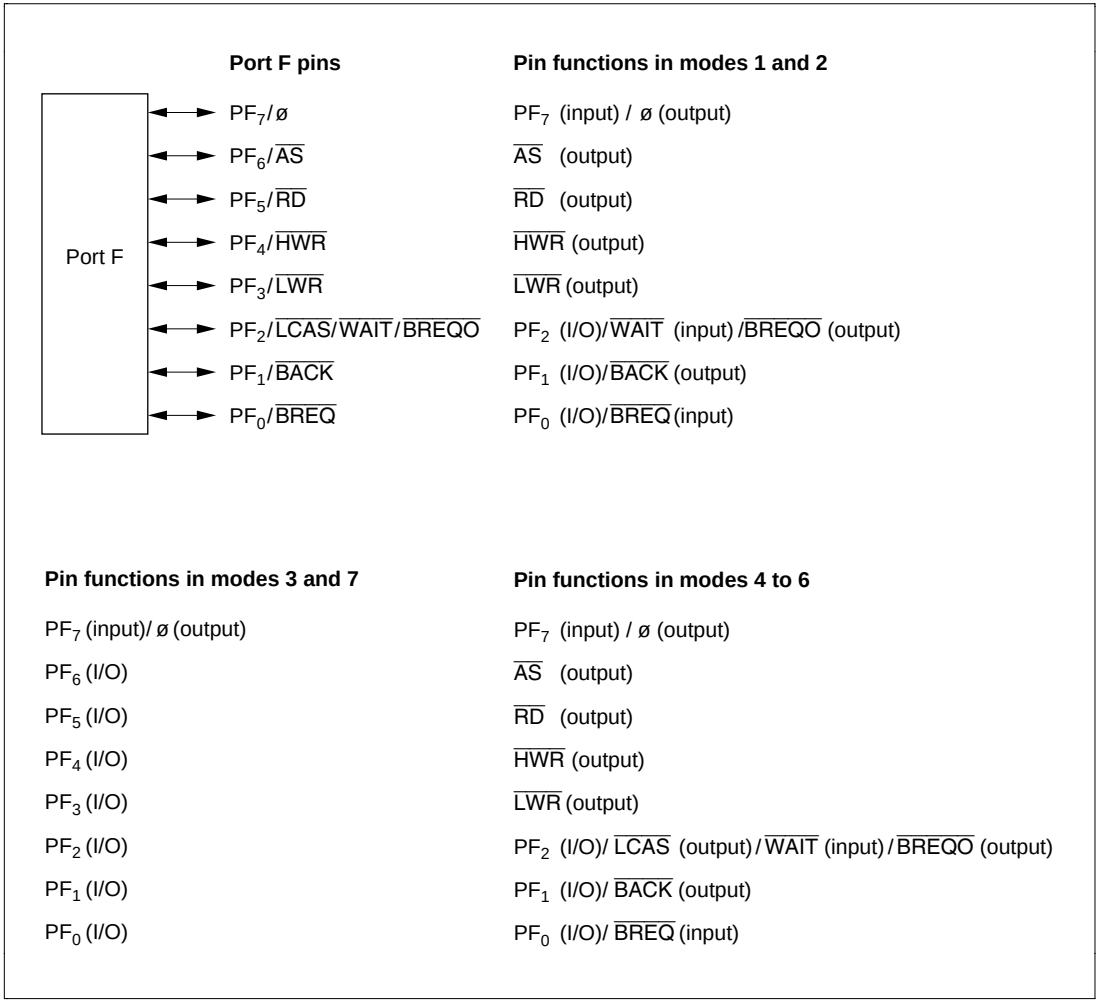



**Figure 9-25 Port F Pin Functions**

### 9.13.2 Register Configuration

Table 9-23 shows the port F register configuration.

**Table 9-23 Port F Registers**

| Name | Abbreviation | R/W | Initial Value | Address *[1] |
|---|---|---|---|---|
| Port F data direction register | PFDDR | W | H'80/H'00*[2] | H'FEBE |
| Port F data register | PFDR | R/W | H'00 | H'FF6E |
| Port F register | PORTF | R | Undefined | H'FF5E |

Notes: 1. Lower 16 bits of the address.
2. Initial value depends on the mode.

**Port F Data Direction Register (PFDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR |
| Modes 1, 2, 4, 5, 6 | | | | | | | | | |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |
| Modes 3 and 7 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PFDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port F. PFDDR cannot be read; if it is, an undefined value will be read.

PFDDR is initialized by a power-on reset, and in hardware standby mode, to H'80 in modes 1, 2, 4, 5, and 6, and to H'00 in modes 3 and 7. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the bus control output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 2, 4, 5, and 6

  Pin $PF_7$ functions as the ø output pin when the corresponding PFDDR bit is set to 1, and as an input port when the bit is cleared to 0.

  The input/output direction specified by PFDDR is ignored for pins $PF_6$ to $PF_3$, which are automatically designated as bus control outputs ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$).

Pins $PF_2$ to $PF_0$ are designated as bus control input/output pins ($\overline{LCAS}$, $\overline{WAIT}$, $\overline{BREQO}$, $\overline{BACK}$, $\overline{BREQ}$) by means of bus controller settings. At other times, setting a PFDDR bit to 1 makes the corresponding port F pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7

  Setting a PFDDR bit to 1 makes the corresponding port F pin $PF_6$ to $PF_0$ an output port, or in the case of pin $PF_7$, the ø output pin. Clearing the bit to 0 makes the pin an input port.

**Port F Data Register (PFDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PFDR is an 8-bit readable/writable register that stores output data for the port F pins ($PF_7$ to $PF_0$).

PFDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port F Register (PORTF)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PF_7$ to $PF_0$.

PORTF is an 8-bit read-only register that shows the pin states. Writing of output data for the port F pins ($PF_7$ to $PF_0$) must always be performed on PFDR.

If a port F read is performed while PFDDR bits are set to 1, the PFDR values are read. If a port F read is performed while PFDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTF contents are determined by the pin states, as PFDDR and PFDR are initialized. PORTF retains its prior state after a manual reset, and in software standby mode.

### 9.13.3 Pin Functions

Port F pins also function as bus control signal input/output pins ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, $\overline{LCAS}$, $\overline{WAIT}$, $\overline{BREQO}$, $\overline{BREQ}$, and $\overline{BACK}$) and the system clock (ø) output pin. The pin functions differ between modes 1, 2, 4, 5, and 6, and modes 3 and 7. Port F pin functions are shown in table 9-24.

**Table 9-24 Port F Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $PF_7$/ø | The pin function is switched as shown below according to bit PF7DDR. |

| PF7DDR | 0 | 1 |
|---|---|---|
| Pin function | $PF_7$ input pin | ø output pin |

$PF_6$/$\overline{AS}$ — The pin function is switched as shown below according to the operating mode and bit PF6DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | Modes 3 and 7 | |
|---|---|---|---|
| PF6DDR | — | 0 | 1 |
| Pin function | $\overline{AS}$ output pin | $PF_6$ input pin | $PF_6$ output pin |

$PF_5$/$\overline{RD}$ — The pin function is switched as shown below according to the operating mode and bit PF5DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | Modes 3 and 7 | |
|---|---|---|---|
| PF5DDR | — | 0 | 1 |
| Pin function | $\overline{RD}$ output pin | $PF_5$ input pin | $PF_5$ output pin |

$PF_4$/$\overline{HWR}$ — The pin function is switched as shown below according to the operating mode and bit PF4DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | Modes 3 and 7 | |
|---|---|---|---|
| PF4DDR | — | 0 | 1 |
| Pin function | $\overline{HWR}$ output pin | $PF_4$ input pin | $PF_4$ output pin |

**Table 9-24 Port F Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|

$PF_3/\overline{LWR}$

The pin function is switched as shown below according to the operating mode and bit PF3DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | Modes 3 and 7 | |
|---|---|---|---|
| PF3DDR | — | 0 | 1 |
| Pin function | $\overline{LWR}$ output pin | $PF_3$ input pin | $PF_3$ output pin |

$PF_2/\overline{LCAS}/\overline{WAIT}/\overline{BREQO}$

The pin function is switched as shown below according to the combination of the operating mode, and bits RMTS2 to RMTS0, LCASS, BREQOE, WAITE, and PF2DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | | | | | Modes 3 and 7 | |
|---|---|---|---|---|---|---|---|
| LCASS | 0 | | | | * | — | |
| BREQOE | 0 | | | 1 | — | — | |
| WAITE | 0 | | 1 | — | — | — | |
| PF2DDR | 0 | 1 | — | — | — | 0 | 1 |
| Pin function | $PF_2$ input pin | $PF_2$ output pin | $\overline{WAIT}$ input pin | $\overline{BREQO}$ output pin | $\overline{LCAS}$ output pin | $PF_2$ input pin | $PF_2$ output pin |

Note: * Only when RMTS2 to RMTS0 = B'001 to B'011 and CW2 = 0 in modes 4 to 6.

$PF_1/\overline{BACK}$

The pin function is switched as shown below according to the combination of the operating mode, and bits BRLE and PF1DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | | | Modes 3 and 7 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF1DDR | 0 | 1 | — | 0 | 1 |
| Pin function | $PF_1$ input pin | $PF_1$ output pin | $\overline{BACK}$ output pin | $PF_1$ input pin | $PF_1$ output pin |

$PF_0/\overline{BREQ}$

The pin function is switched as shown below according to the combination of the operating mode, and bits BRLE and PF0DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | | | Modes 3 and 7 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF0DDR | 0 | 1 | — | 0 | 1 |
| Pin function | $PF_0$ input pin | $PF_0$ output pin | $\overline{BREQ}$ input pin | $PF_0$ input pin | $PF_0$ output pin |

## 9.14 Port G

### 9.14.1 Overview

Port G is a 5-bit I/O port. Port G pins also function as bus control signal output pins ($\overline{CS_0}$ to $\overline{CS_3}$, $\overline{CAS}$, and $\overline{OE}$).

Figure 9-26 shows the port G pin configuration.

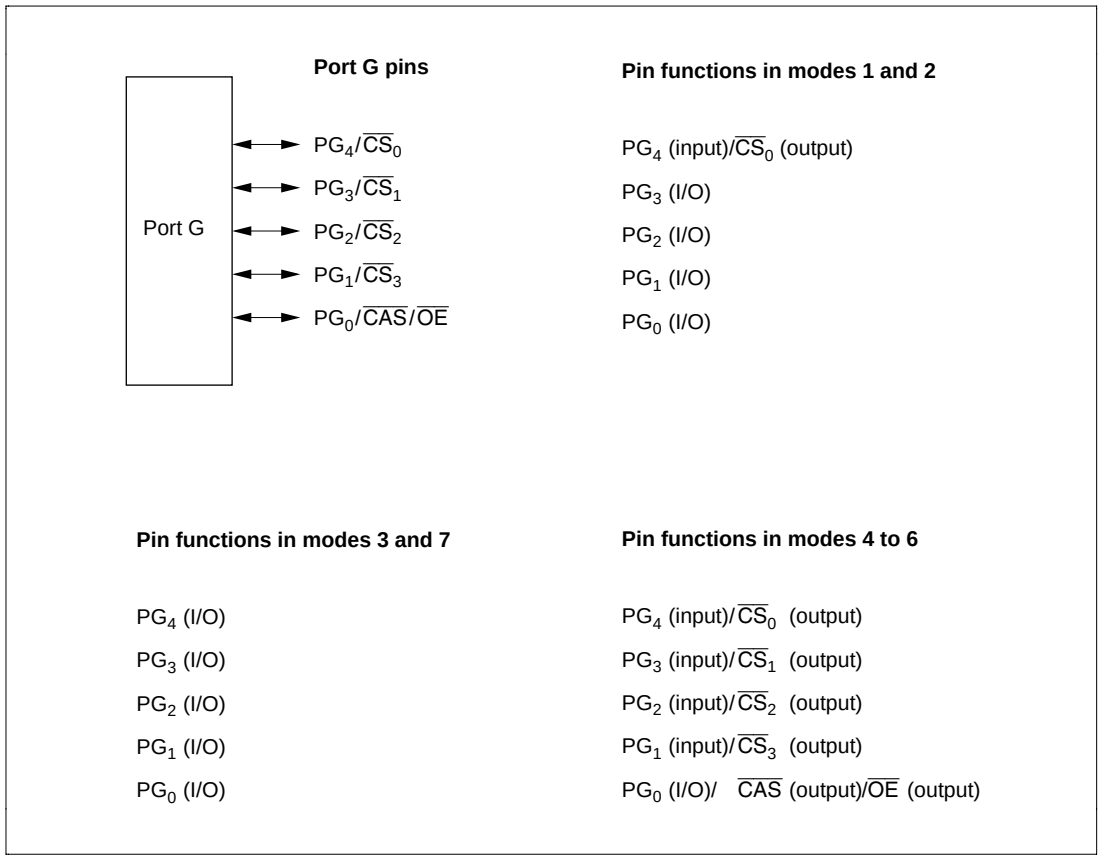



**Figure 9-26 Port G Pin Functions**

### 9.14.2 Register Configuration

Table 9-25 shows the port G register configuration.

**Table 9-25 Port G Registers**

| Name | Abbreviation | R/W | Initial Value | Address *[1] |
|---|---|---|---|---|
| Port G data direction register | PGDDR | W | H'E0/H'F0*[2] | H'FEBF |
| Port G data register | PGDR | R/W | H'00 | H'FF6F |
| Port G register | PORTG | R | Undefined | H'FF5F |

Notes: 1. Lower 16 bits of the address.
2. Initial value depends on the mode.

**Port G Data Direction Register (PGDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR |
| Modes 1, 4, 5 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | W | W | W | W | W |
| Modes 2, 3, 6, 7 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | W | W | W | W | W |

PGDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port G. PGDDR cannot be read, and bits 7 to 5 are reserved. If PGDDR is read, an undefined value will be read.

PGDDR is initialized by a power-on reset, and in hardware standby mode, to H'F0 in modes 1, 4, and 5, and to H'E0 in modes 2, 3, 6, and 7. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the bus control output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1 and 2

  Pin $PG_4$ functions as a bus control output pin ($\overline{CS}_0$) when the corresponding PGDDR bit is set to 1, and as an input port when the bit is cleared to 0.

  For pins $PG_3$ to $PG_0$, setting the corresponding PGDDR bit to 1 makes the pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7

  Setting a PGDDR bit to 1 makes the corresponding port G pin an output port, while clearing the bit to 0 makes the pin an input port.
- Modes 4, 5, and 6

  Pins $PG_4$ to $PG_1$ function as bus control output pins ($\overline{CS_0}$ to $\overline{CS_3}$) when the corresponding PGDDR bits are set to 1, and as input ports when the bits are cleared to 0.

  Pin $PG_0$ functions as the $\overline{CAS}$ output pin when DRAM interface is designated, and as the $\overline{OE}$ output pin when PSRAM interface is designated. Otherwise, setting the corresponding PGDDR bit to 1 makes the pin an output port, while clearing the bit to 0 makes the pin an input port.

  For details of the DRAM and PSRAM interfaces, see section 6, Bus Controller.

**Port G Data Register (PGDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR |
| Initial value | : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | R/W | R/W | R/W | R/W | R/W |

PGDR is an 8-bit readable/writable register that stores output data for the port G pins ($PG_4$ to $PG_0$).

Bits 7 to 5 are reserved; they return an undetermined value if read, and cannot be modified.

PGDR is initialized to H'E0 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port G Register (PORTG)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 |
| Initial value | : | 1 | 1 | 1 | —* | —* | —* | —* | —* |
| R/W | : | — | — | — | R | R | R | R | R |

Note: * Determined by state of pins $PG_4$ to $PG_0$.

PORTG is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port G pins ($PG_4$ to $PG_0$) must always be performed on PGDR.

Bits 7 to 5 are reserved; they return an undetermined value if read, and cannot be modified.

If a port G read is performed while PGDDR bits are set to 1, the PGDR values are read. If a port G read is performed while PGDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTG contents are determined by the pin states, as PGDDR and PGDR are initialized. PORTG retains its prior state after a manual reset, and in software standby mode.

### 9.14.3 Pin Functions

Port G pins also function as bus control signal output pins ($\overline{CS_0}$ to $\overline{CS_3}$, $\overline{CAS}$, and $\overline{OE}$). The pin functions are different in modes 1 and 2, modes 3 and 7, and modes 4 to 6. Port G pin functions are shown in table 9-26.

**Table 9-26 Port G Pin Functions**

**Pin** | **Selection Method and Pin Functions**

$PG_4/\overline{CS_0}$

The pin function is switched as shown below according to the operating mode and bit PG4DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6 | | Modes 3 and 7 | |
|---|---|---|---|---|
| PG4DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_4$ input pin | $\overline{CS_0}$ output pin | $PG_4$ input pin | $PG_4$ output pin |

$PG_3/\overline{CS_1}$

The pin function is switched as shown below according to the operating mode and bit PG3DDR.

| Operating Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| PG3DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_3$ input pin | $PG_3$ output pin | $PG_3$ input pin | $\overline{CS_1}$ output pin |

$PG_2/\overline{CS_2}$

The pin function is switched as shown below according to the operating mode and bit PG2DDR.

| Operating Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| PG2DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_2$ input pin | $PG_2$ output pin | $PG_2$ input pin | $\overline{CS_2}$ output pin |

**Table 9-26 Port G Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $PG_1/\overline{CS}_3$ | The pin function is switched as shown below according to the operating mode and bit PG1DDR. |

| Operating Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | |
|---|---|---|---|---|
| PG1DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_1$ input pin | $PG_1$ output pin | $PG_1$ input pin | $\overline{CS}_3$ output pin |

| $PG_0/\overline{CAS}/\overline{OE}$ | The pin function is switched as shown below according to the combination of the operating mode and bits RMTS2 to RMTS0 and PG0DDR. |
|---|---|

| Operating Mode | Modes 1, 2, 3, 7 | | Modes 4 to 6 | | | |
|---|---|---|---|---|---|---|
| RMTS2 to RMTS0 | — | | B'000, B'100 | | B'0XX | B'1XX |
| PG0DDR | 0 | 1 | 0 | 1 | — | |
| Pin function | $PG_0$ input pin | $PG_0$ output pin | $PG_0$ input pin | $PG_0$ output pin | $\overline{CAS}$ output | $\overline{OE}$ output |

XX ≠ 00

# Section 10 16-Bit Timer Pulse Unit (TPU)

## 10.1 Overview

The H8S/2655 Series has an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

### 10.1.1 Features

- Maximum 16-pulse input/output
- A total of 16 timer general registers (TGRs) are provided (four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5), each of which can be set independently as an output compare/input capture register
  TGRC and TGRD for channels 0 and 3 can also be used as buffer registers
- Selection of 8 counter input clocks for each channel
- The following operations can be set for each channel:
  — Waveform output at compare match: Selection of 0, 1, or toggle output
  — Input capture function: Selection of rising edge, falling edge, or both edge detection
  — Counter clear operation: Counter clearing possible by compare match or input capture
  — Synchronous operation: Multiple timer counters (TCNT) can be written to simultaneously
  Simultaneous clearing by compare match and input capture possible
  Register simultaneous input/output possible by counter synchronous operation
  — PWM mode: Any PWM output duty can be set
  Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
  — Input capture register double-buffering possible
  — Automatic rewriting of output compare register possible
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
  — Two-phase encoder pulse up/down-count possible
- Cascaded operation
  — Channel 2 (channel 5) input clock operates as 32-bit counter by setting channel 1 (channel 4) overflow/underflow

- Fast access via internal 16-bit bus
  - Fast access is possible via a 16-bit interface
- 26 interrupt sources
  - For channels 0 and 3, four compare match/input capture dual-function interrupts and one overflow interrupt can be requested independently
  - For channels 1, 2, 4, and 5, two compare match/input capture dual-function interrupts, one overflow interrupt, and one underflow interrupt can be requested independently
- Automatic transfer of register data
  - Block transfer, 1-word data transfer, and 1-byte data transfer possible by data transfer controller (DTC) or DMA controller (DMAC) activation
- Programmable pulse generator (PPG) output trigger can be generated
  - Channel 0 to 3 compare match/input capture signals can be used as PPG output trigger
- A/D converter conversion start trigger can be generated
  - Channel 0 to 5 compare match A/input capture A signals can be used as A/D converter conversion start trigger

Table 10-1 lists the functions of the TPU.

**Table 10-1 TPU Functions**

| Item | | Channel 0 | Channel 1 | Channel 2 | Channel 3 | Channel 4 | Channel 5 |
|---|---|---|---|---|---|---|---|
| Count clock | | ø/1<br>ø/4<br>ø/16<br>ø/64<br>TCLKA<br>TCLKB<br>TCLKC<br>TCLKD | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>TCLKA<br>TCLKB | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/1024<br>TCLKA<br>TCLKB<br>TCLKC | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>ø/1024<br>ø/4096<br>TCLKA | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/1024<br>TCLKA<br>TCLKC | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>TCLKA<br>TCLKC<br>TCLKD |
| General registers | | TGR0A<br>TGR0B | TGR1A<br>TGR1B | TGR2A<br>TGR2B | TGR3A<br>TGR3B | TGR4A<br>TGR4B | TGR5A<br>TGR5B |
| General registers/ buffer registers | | TGR0C<br>TGR0D | — | — | TGR3C<br>TGR3D | — | — |
| I/O pins | | TIOCA0<br>TIOCB0<br>TIOCC0<br>TIOCD0 | TIOCA1<br>TIOCB1 | TIOCA2<br>TIOCB2 | TIOCA3<br>TIOCB3<br>TIOCC3<br>TIOCD3 | TIOCA4<br>TIOCB4 | TIOCA5<br>TIOCB5 |
| Counter clear function | | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture |
| Compare match output | 0 output | ○ | ○ | ○ | ○ | ○ | ○ |
| | 1 output | ○ | ○ | ○ | ○ | ○ | ○ |
| | Toggle output | ○ | ○ | ○ | ○ | ○ | ○ |
| Input capture function | | ○ | ○ | ○ | ○ | ○ | ○ |
| Synchronous operation | | ○ | ○ | ○ | ○ | ○ | ○ |
| PWM mode | | ○ | ○ | ○ | ○ | ○ | ○ |
| Phase counting mode | | — | ○ | ○ | — | ○ | ○ |
| Buffer operation | | ○ | — | — | ○ | — | — |

Legend

○ : Possible

— : Not possible

**Table 10-1 TPU Functions (cont)**

| Item | Channel 0 | Channel 1 | Channel 2 | Channel 3 | Channel 4 | Channel 5 |
|---|---|---|---|---|---|---|
| DMAC activation | TGR0A compare match or input capture | TGR1A compare match or input capture | TGR2A compare match or input capture | TGR3A compare match or input capture | TGR4A compare match or input capture | TGR5A compare match or input capture |
| DTC activation | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture |
| A/D converter trigger | TGR0A compare match or input capture | TGR1A compare match or input capture | TGR2A compare match or input capture | TGR3A compare match or input capture | TGR4A compare match or input capture | TGR5A compare match or input capture |
| PPG trigger | TGR0A/ TGR0B compare match or input capture | TGR1A/ TGR1B compare match or input capture | TGR2A/ TGR2B compare match or input capture | TGR3A/ TGR3B compare match or input capture | — | — |
| Interrupt sources | 5 sources<br>• Compare match or input capture 0A<br>• Compare match or input capture 0B<br>• Compare match or input capture 0C<br>• Compare match or input capture 0D<br>• Overflow | 4 sources<br>• Compare match or input capture 1A<br>• Compare match or input capture 1B<br>• Overflow<br>• Underflow | 4 sources<br>• Compare match or input capture 2A<br>• Compare match or input capture 2B<br>• Overflow<br>• Underflow | 5 sources<br>• Compare match or input capture 3A<br>• Compare match or input capture 3B<br>• Compare match or input capture 3C<br>• Compare match or input capture 3D<br>• Overflow | 4 sources<br>• Compare match or input capture 4A<br>• Compare match or input capture 4B<br>• Overflow<br>• Underflow | 4 sources<br>• Compare match or input capture 5A<br>• Compare match or input capture 5B<br>• Overflow<br>• Underflow |

Legend

— : Not possible

### 10.1.2 Block Diagram

Figure 10-1 shows a block diagram of the TPU.

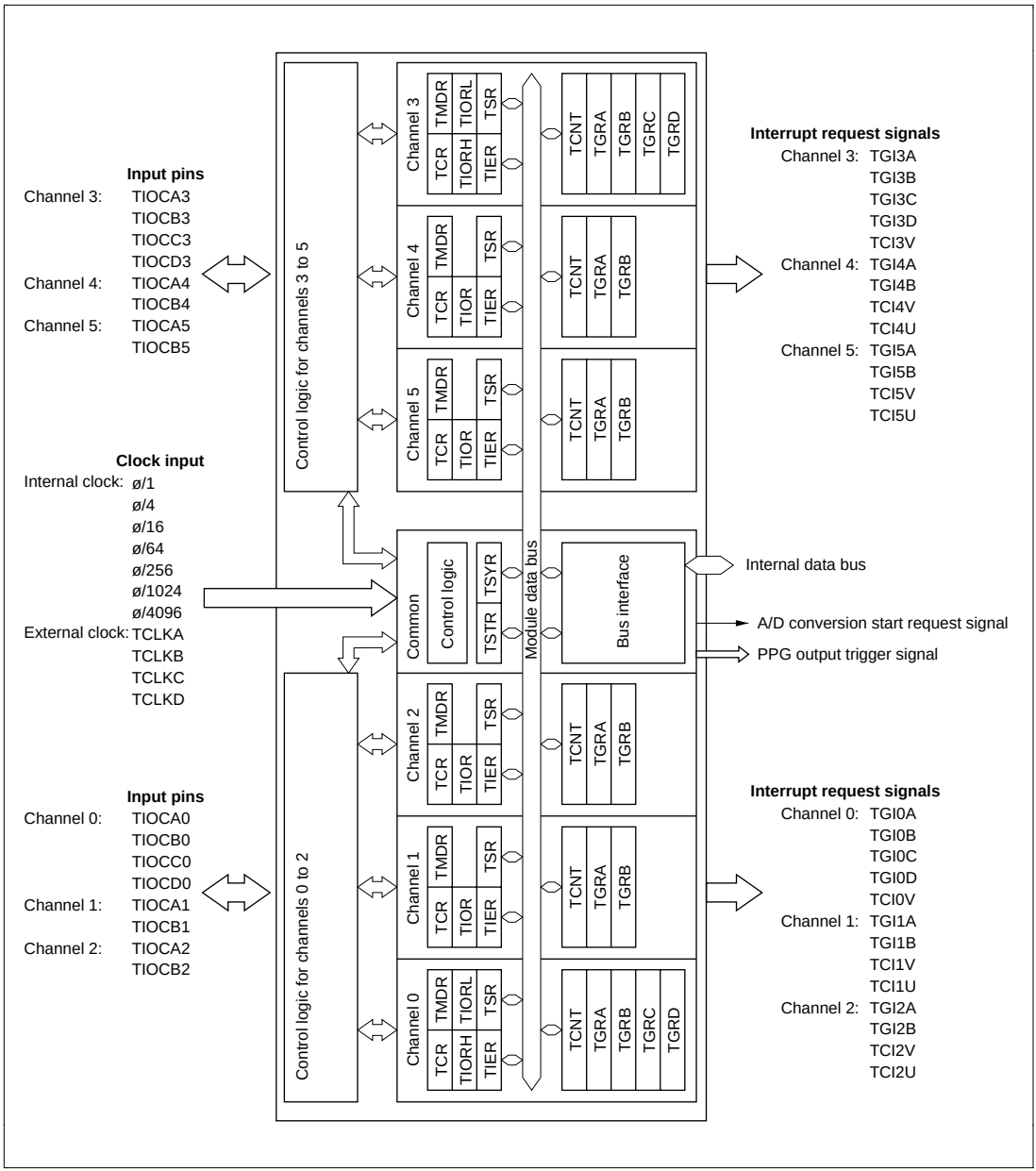



**Figure 10-1 Block Diagram of TPU**

### 10.1.3 Pin Configuration

Table 10-2 summarizes the TPU pins.

**Table 10-2 TPU Pins**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| All | Clock input A | TCLKA | Input | External clock A input pin<br>(Channel 1 and 5 phase counting mode A phase input) |
| | Clock input B | TCLKB | Input | External clock B input pin<br>(Channel 1 and 5 phase counting mode B phase input) |
| | Clock input C | TCLKC | Input | External clock C input pin<br>(Channel 2 and 4 phase counting mode A phase input) |
| | Clock input D | TCLKD | Input | External clock D input pin<br>(Channel 2 and 4 phase counting mode B phase input) |
| 0 | Input capture/out compare match A0 | TIOCA0 | I/O | TGR0A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B0 | TIOCB0 | I/O | TGR0B input capture input/output compare output/PWM output pin |
| | Input capture/out compare match C0 | TIOCC0 | I/O | TGR0C input capture input/output compare output/PWM output pin |
| | Input capture/out compare match D0 | TIOCD0 | I/O | TGR0D input capture input/output compare output/PWM output pin |
| 1 | Input capture/out compare match A1 | TIOCA1 | I/O | TGR1A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B1 | TIOCB1 | I/O | TGR1B input capture input/output compare output/PWM output pin |
| 2 | Input capture/out compare match A2 | TIOCA2 | I/O | TGR2A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B2 | TIOCB2 | I/O | TGR2B input capture input/output compare output/PWM output pin |

**Table 10-2 TPU Pins (cont)**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 3 | Input capture/out compare match A3 | TIOCA3 | I/O | TGR3A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B3 | TIOCB3 | I/O | TGR3B input capture input/output compare output/PWM output pin |
| | Input capture/out compare match C3 | TIOCC3 | I/O | TGR3C input capture input/output compare output/PWM output pin |
| | Input capture/out compare match D3 | TIOCD3 | I/O | TGR3D input capture input/output compare output/PWM output pin |
| 4 | Input capture/out compare match A4 | TIOCA4 | I/O | TGR4A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B4 | TIOCB4 | I/O | TGR4B input capture input/output compare output/PWM output pin |
| 5 | Input capture/out compare match A5 | TIOCA5 | I/O | TGR5A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B5 | TIOCB5 | I/O | TGR5B input capture input/output compare output/PWM output pin |

### 10.1.4 Register Configuration

Table 10-3 summarizes the TPU registers.

**Table 10-3 TPU Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address *1 |
|---|---|---|---|---|---|
| 0 | Timer control register 0 | TCR0 | R/W | H'00 | H'FFD0 |
| | Timer mode register 0 | TMDR0 | R/W | H'C0 | H'FFD1 |
| | Timer I/O control register 0H | TIOR0H | R/W | H'00 | H'FFD2 |
| | Timer I/O control register 0L | TIOR0L | R/W | H'00 | H'FFD3 |
| | Timer interrupt enable register 0 | TIER0 | R/W | H'40 | H'FFD4 |
| | Timer status register 0 | TSR0 | R/(W)*2 | H'C0 | H'FFD5 |
| | Timer counter 0 | TCNT0 | R/W | H'0000 | H'FFD6 |
| | Timer general register 0A | TGR0A | R/W | H'FFFF | H'FFD8 |
| | Timer general register 0B | TGR0B | R/W | H'FFFF | H'FFDA |
| | Timer general register 0C | TGR0C | R/W | H'FFFF | H'FFDC |
| | Timer general register 0D | TGR0D | R/W | H'FFFF | H'FFDE |
| 1 | Timer control register 1 | TCR1 | R/W | H'00 | H'FFE0 |
| | Timer mode register 1 | TMDR1 | R/W | H'C0 | H'FFE1 |
| | Timer I/O control register 1 | TIOR1 | R/W | H'00 | H'FFE2 |
| | Timer interrupt enable register 1 | TIER1 | R/W | H'40 | H'FFE4 |
| | Timer status register 1 | TSR1 | R/(W) *2 | H'C0 | H'FFE5 |
| | Timer counter 1 | TCNT1 | R/W | H'0000 | H'FFE6 |
| | Timer general register 1A | TGR1A | R/W | H'FFFF | H'FFE8 |
| | Timer general register 1B | TGR1B | R/W | H'FFFF | H'FFEA |
| 2 | Timer control register 2 | TCR2 | R/W | H'00 | H'FFF0 |
| | Timer mode register 2 | TMDR2 | R/W | H'C0 | H'FFF1 |
| | Timer I/O control register 2 | TIOR2 | R/W | H'00 | H'FFF2 |
| | Timer interrupt enable register 2 | TIER2 | R/W | H'40 | H'FFF4 |
| | Timer status register 2 | TSR2 | R/(W) *2 | H'C0 | H'FFF5 |
| | Timer counter 2 | TCNT2 | R/W | H'0000 | H'FFF6 |
| | Timer general register 2A | TGR2A | R/W | H'FFFF | H'FFF8 |
| | Timer general register 2B | TGR2B | R/W | H'FFFF | H'FFFA |

**Table 10-3 TPU Registers (cont)**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 3 | Timer control register 3 | TCR3 | R/W | H'00 | H'FE80 |
| | Timer mode register 3 | TMDR3 | R/W | H'C0 | H'FE81 |
| | Timer I/O control register 3H | TIOR3H | R/W | H'00 | H'FE82 |
| | Timer I/O control register 3L | TIOR3L | R/W | H'00 | H'FE83 |
| | Timer interrupt enable register 3 | TIER3 | R/W | H'40 | H'FE84 |
| | Timer status register 3 | TSR3 | R/(W)*[2] | H'C0 | H'FE85 |
| | Timer counter 3 | TCNT3 | R/W | H'0000 | H'FE86 |
| | Timer general register 3A | TGR3A | R/W | H'FFFF | H'FE88 |
| | Timer general register 3B | TGR3B | R/W | H'FFFF | H'FE8A |
| | Timer general register 3C | TGR3C | R/W | H'FFFF | H'FE8C |
| | Timer general register 3D | TGR3D | R/W | H'FFFF | H'FE8E |
| 4 | Timer control register 4 | TCR4 | R/W | H'00 | H'FE90 |
| | Timer mode register 4 | TMDR4 | R/W | H'C0 | H'FE91 |
| | Timer I/O control register 4 | TIOR4 | R/W | H'00 | H'FE92 |
| | Timer interrupt enable register 4 | TIER4 | R/W | H'40 | H'FE94 |
| | Timer status register 4 | TSR4 | R/(W) *[2] | H'C0 | H'FE95 |
| | Timer counter 4 | TCNT4 | R/W | H'0000 | H'FE96 |
| | Timer general register 4A | TGR4A | R/W | H'FFFF | H'FE98 |
| | Timer general register 4B | TGR4B | R/W | H'FFFF | H'FE9A |
| 5 | Timer control register 5 | TCR5 | R/W | H'00 | H'FEA0 |
| | Timer mode register 5 | TMDR5 | R/W | H'C0 | H'FEA1 |
| | Timer I/O control register 5 | TIOR5 | R/W | H'00 | H'FEA2 |
| | Timer interrupt enable register 5 | TIER5 | R/W | H'40 | H'FEA4 |
| | Timer status register 5 | TSR5 | R/(W) *[2] | H'C0 | H'FEA5 |
| | Timer counter 5 | TCNT5 | R/W | H'0000 | H'FEA6 |
| | Timer general register 5A | TGR5A | R/W | H'FFFF | H'FEA8 |
| | Timer general register 5B | TGR5B | R/W | H'FFFF | H'FEAA |
| All | Timer start register | TSTR | R/W | H'00 | H'FFC0 |
| | Timer synchro register | TSYR | R/W | H'00 | H'FFC1 |
| | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 10.2 Register Descriptions

### 10.2.1 Timer Control Register (TCR)

The TCR registers are 8-bit registers that control the TCNT channels. The TPU has six TCR registers, one for each of channels 0 to 5. The TCR registers are initialized to H'00 by a reset, and in hardware standby mode. TCR register settings should be made only when TCNT operation is stopped.

**Channel 0: TCR0**

**Channel 3: TCR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TCR1**

**Channel 2: TCR2**

**Channel 4: TCR4**

**Channel 5: TCR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7, 6, 5—Counter Clear 2, 1, and 0 (CCLR2, CCLR1, CCLR0):** These bits select the TCNT counter clearing source.

| Channel | Bit 7 CCLR2 | Bit 6 CCLR1 | Bit 5 CCLR0 | Description |
|---|---|---|---|---|
| 0, 3 | 0 | 0 | 0 | TCNT clearing disabled (Initial value) |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |
| | 1 | 0 | 0 | TCNT clearing disabled |
| | | | 1 | TCNT cleared by TGRC compare match/input capture *2 |
| | | 1 | 0 | TCNT cleared by TGRD compare match/input capture *2 |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |

| Channel | Bit 7 Reserved*3 | Bit 6 CCLR1 | Bit 5 CCLR0 | Description |
|---|---|---|---|---|
| 1, 2, 4, 5 | 0 | 0 | 0 | TCNT clearing disabled (Initial value) |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.
3. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

**Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0):** These bits select the input clock edge. When the input clock is counted using both edges, the input clock period is halved (e.g. ø/4 both edges = ø/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority.

| Bit 4 | Bit 3 | | |
|---|---|---|---|
| **CKEG1** | **CKEG0** | **Description** | |
| 0 | 0 | Count at rising edge | (Initial value) |
| | 1 | Count at falling edge | |
| 1 | * | Count at both edges | |

*: Don't care

Note: Internal clock edge selection is valid when the input clock is ø/4 or slower. This setting is ignored if the input clock is ø/1, or when overflow/underflow of another channel is selected.

**Bits 2, 1, and 0—Time Prescaler 2, 1, and 0 (TPSC2 to TPSC0):** These bits select the TCNT counter clock. The clock source can be selected independently for each channel. Table 10-4 shows the clock sources that can be set for each channel.

**Table 10-4 TPU Clock Sources**

| | Internal Clock | | | | | | | External Clock | | | | Overflow/ Underflow on Another |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Channel** | **ø/1** | **ø/4** | **ø/16** | **ø/64** | **ø/256** | **ø/1024** | **ø/4096** | **TCLKA** | **TCLKB** | **TCLKC** | **TCLKD** | **Channel** |
| 0 | ○ | ○ | ○ | ○ | | | | ○ | ○ | ○ | ○ | |
| 1 | ○ | ○ | ○ | ○ | ○ | | | ○ | ○ | | | ○ |
| 2 | ○ | ○ | ○ | ○ | | ○ | | ○ | ○ | ○ | | |
| 3 | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | | | | |
| 4 | ○ | ○ | ○ | ○ | | ○ | | ○ | | ○ | | ○ |
| 5 | ○ | ○ | ○ | ○ | ○ | | | ○ | | ○ | ○ | |

**Legend**

○ : Setting

Blank : No setting

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | External clock: counts on TCLKC pin input | |
| | | | 1 | External clock: counts on TCLKD pin input | |

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | Counts on TCNT2 overflow/underflow | |

Note: This setting is ignored when channel 1 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | External clock: counts on TCLKC pin input | |
| | | | 1 | Internal clock: counts on ø/1024 | |

Note: This setting is ignored when channel 2 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | Internal clock: counts on ø/1024 | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | Internal clock: counts on ø/4096 | |

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKC pin input | |
| | | 1 | 0 | Internal clock: counts on ø/1024 | |
| | | | 1 | Counts on TCNT5 overflow/underflow | |

Note: This setting is ignored when channel 4 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKC pin input | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | External clock: counts on TCLKD pin input | |

Note: This setting is ignored when channel 5 is in phase counting mode.

### 10.2.2 Timer Mode Register (TMDR)

The TMDR registers are 8-bit readable/writable registers that are used to set the operating mode for each channel. The TPU has six TMDR registers, one for each channel. The TMDR registers are initialized to H'C0 by a reset, and in hardware standby mode. TMDR register settings should be made only when TCNT operation is stopped.

**Channel 0: TMDR0**

**Channel 3: TMDR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TMDR1**

**Channel 2: TMDR2**

**Channel 4: TMDR4**

**Channel 5: TMDR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Reserved:** Read-only bits, always read as 1.

**Bit 5—Buffer Operation B (BFB):** Specifies whether TGRB is to operate in the normal way, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

| BFB | Description | |
|---|---|---|
| 0 | TGRB operates normally | (Initial value) |
| 1 | TGRB and TGRD used together for buffer operation | |

**Bit 4—Buffer Operation A (BFA):** Specifies whether TGRA is to operate in the normal way, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified.

**Bit 4**

| BFA | Description | |
|---|---|---|
| 0 | TGRA operates normally | (Initial value) |
| 1 | TGRA and TGRC used together for buffer operation | |

**Bits 3 to 0—Modes 3 to 0 (MD3 to MD0):** These bits are used to set the timer operating mode.

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|---|
| MD3*[1] | MD2*[2] | MD1 | MD0 | Description | |
| 0 | 0 | 0 | 0 | Normal operation | (Initial value) |
| | | | 1 | Reserved | |
| | | 1 | 0 | PWM mode 1 | |
| | | | 1 | PWM mode 2 | |
| | 1 | 0 | 0 | Phase counting mode 1 | |
| | | | 1 | Phase counting mode 2 | |
| | | 1 | 0 | Phase counting mode 3 | |
| | | | 1 | Phase counting mode 4 | |
| 1 | * | * | * | — | |

*: Don’t care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

### 10.2.3 Timer I/O Control Register (TIOR)

**Channel 0: TIOR0H**

**Channel 1: TIOR1**

**Channel 2: TIOR2**

**Channel 3: TIOR3H**

**Channel 4: TIOR4**

**Channel 5: TIOR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 0: TIOR0L**

**Channel 3: TIOR3L**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

The TIOR registers are 8-bit registers that control the TGR registers. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. The TIOR registers are initialized to H'00 by a reset, and in hardware standby mode.

Care is required since TIOR is affected by the TMDR setting. The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

**Bits 7 to 4— I/O Control B3 to B0 (IOB3 to IOB0)**
**I/O Control D3 to D0 (IOD3 to IOD0):**

Bits IOB3 to IOB0 specify the function of TGRB.
Bits IOD3 to IOD0 specify the function of TGRD.

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | | | |
|---|---|---|---|---|---|---|---|
| **Channel** | **IOB3** | **IOB2** | **IOB1** | **IOB0** | **Description** | | |
| 0 | 0 | 0 | 0 | 0 | TGR0B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0B is input capture register | Capture input source is TIOCB0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count- up/count-down*[1] |

*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

| Channel | Bit 7 IOD3 | Bit 6 IOD2 | Bit 5 IOD1 | Bit 4 IOD0 | Description | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | TGR0D is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0D is input capture register | Capture input source is TIOCD0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/count-down*[1] |

*: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR0 is set to 1 and TGR0D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | TGR1B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR1B is input capture register | Capture input source is TIOCB1 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR0C compare match/ input capture | Input capture at generation of TGR0C compare match/input capture |

*: Don’t care

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | TGR2B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR2B is input capture register | Capture input source is TIOCB2 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3B is input capture register | Capture input source is TIOCB3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down*[1] |

*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

| Channel | Bit 7 IOD3 | Bit 6 IOD2 | Bit 5 IOD1 | Bit 4 IOD0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3D is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3D is input capture register | Capture input source is TIOCD3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down*[1] |

*: Don’t care

Notes: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR3 is set to 1 and TGR3D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | 0 | TGR4B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR4B is input capture register | Capture input source is TIOCB4 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR3C compare match/ input capture | Input capture at generation of TGR3C compare match/ input capture |

*: Don't care

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | 0 | TGR5B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR5B is input capture register | Capture input source is TIOCB5 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don't care

**Bits 3 to 0— I/O Control A3 to A0 (IOA3 to IOA0)**
**I/O Control C3 to C0 (IOC3 to IOC0):**

IOA3 to IOA0 specify the function of TGRA.
IOC3 to IOC0 specify the function of TGRC.

| | Bit 3 | Bit 2 | Bit 1 | Bit 0 | | | |
|---|---|---|---|---|---|---|---|
| **Channel** | **IOA3** | **IOA2** | **IOA1** | **IOA0** | **Description** | | |
| 0 | 0 | 0 | 0 | 0 | TGR0A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0A is input capture register | Capture input source is TIOCA0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/ count clock | Input capture at TCNT1 count-up/count-down |

*: Don’t care

| Channel | Bit 3 IOC3 | Bit 2 IOC2 | Bit 1 IOC1 | Bit 0 IOC0 | Description | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | TGR0C is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0C is input capture register | Capture input source is TIOCC0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/count-down |

*: Don't care

Note: 1. When the BFA bit in TMDR0 is set to 1 and TGR0C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | TGR1A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR1A is input capture register | Capture input source is TIOCA1 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR0A compare match/ input capture | Input capture at generation of channel 0/TGR0A compare match/input capture |

*: Don’t care

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | TGR2A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR2A is input capture register | Capture input source is TIOCA2 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3A is input capture register | Capture input source is TIOCA3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down |

*: Don't care

| Channel | Bit 3 IOC3 | Bit 2 IOC2 | Bit 1 IOC1 | Bit 0 IOC0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3C is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3C is input capture register | Capture input source is TIOCC3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down |

*: Don't care

Note: 1. When the BFA bit in TMDR3 is set to 1 and TGR3C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | 0 | TGR4A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR4A is input capture register | Capture input source is TIOCA4 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR3A compare match/ input capture | Input capture at generation of TGR3A compare match/input capture |

*: Don't care

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | 0 | TGR5A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR5A is input capture register | Capture input source is TIOCA5 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don't care

### 10.2.4 Timer Interrupt Enable Register (TIER)

**Channel 0: TIER0**

**Channel 3: TIER3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value | : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TIER1**

**Channel 2: TIER2**

**Channel 4: TIER4**

**Channel 5: TIER5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | R/W | R/W | — | — | R/W | R/W |

The TIER registers are 8-bit registers that control enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel. The TIER registers are initialized to H'40 by a reset, and in hardware standby mode.

**Bit 7—A/D Conversion Start Request Enable (TTGE):** Enables or disables generation of A/D conversion start requests by TGRA input capture/compare match.

**Bit 7**

| TTGE | Description | |
|---|---|---|
| 0 | A/D conversion start request generation disabled | (Initial value) |
| 1 | A/D conversion start request generation enabled | |

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Interrupt Enable (TCIEU):** Enables or disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1 and 2.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

| TCIEU | Description | |
|---|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled | (Initial value) |
| 1 | Interrupt requests (TCIU) by TCFU enabled | |

**Bit 4—Overflow Interrupt Enable (TCIEV):** Enables or disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1.

**Bit 4**

| TCIEV | Description | |
|---|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled | (Initial value) |
| 1 | Interrupt requests (TCIV) by TCFV enabled | |

**Bit 3—TGR Interrupt Enable D (TGIED):** Enables or disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

**Bit 3**

| TGIED | Description | |
|---|---|---|
| 0 | Interrupt requests (TGID) by TGFD bit disabled | (Initial value) |
| 1 | Interrupt requests (TGID) by TGFD bit enabled | |

**Bit 2—TGR Interrupt Enable C (TGIEC):** Enables or disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

**Bit 2**

| TGIEC | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled | |

**Bit 1—TGR Interrupt Enable B (TGIEB):** Enables or disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.

**Bit 1**

| TGIEB | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled | |

**Bit 0—TGR Interrupt Enable A (TGIEA):** Enables or disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.

**Bit 1**

| TGIEA | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled | |

### 10.2.5 Timer Status Register (TSR)

**Channel 0: TSR0**

**Channel 3: TSR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Can only be written with 0 for flag clearing.

**Channel 1: TSR1**

**Channel 2: TSR2**

**Channel 4: TSR4**

**Channel 5: TSR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Note: * Can only be written with 0 for flag clearing.

The TSR registers are 8-bit registers that indicate the status of each channel. The TPU has six TSR registers, one for each channel. The TSR registers are initialized to H'C0 by a reset, and in hardware standby mode.

**Bit 7—Count Direction Flag (TCFD):** Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5.

In channels 0 and 3, bit 7 is reserved. It is always read as 1 and cannot be modified.

**Bit 7**

| TCFD | Description | |
|---|---|---|
| 0 | TCNT counts down | |
| 1 | TCNT counts up | (Initial value) |

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Flag (TCFU):** Status flag that indicates that TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

| TCFU | Description | |
|---|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 | (Initial value) |
| 1 | [Setting condition]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) | |

**Bit 4—Overflow Flag (TCFV):** Status flag that indicates that TCNT overflow has occurred.

**Bit 4**

| TCFV | Description | |
|---|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 | (Initial value) |
| 1 | [Setting condition]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) | |

**Bit 3—Input Capture/Output Compare Flag D (TGFD):** Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

**Bit 3**

| TGFD | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register |

**Bit 2—Input Capture/Output Compare Flag C (TGFC):** Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

**Bit 2**

| TGFC | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

**Bit 1—Input Capture/Output Compare Flag B (TGFB):** Status flag that indicates the occurrence of TGRB input capture or compare match.

**Bit 1**

| TGFB | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

**Bit 0—Input Capture/Output Compare Flag A (TGFA):** Status flag that indicates the occurrence of TGRA input capture or compare match.

**Bit 0**

| TGFA | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

### 10.2.6 Timer Counter (TCNT)

**Channel 0: TCNT0 (up-counter)**

**Channel 1: TCNT1 (up/down-counter*)**

**Channel 2: TCNT2 (up/down-counter*)**

**Channel 3: TCNT3 (up-counter)**

**Channel 4: TCNT4 (up/down-counter*)**

**Channel 5: TCNT5 (up/down-counter*)**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note : * These counters can be used as up/down-counters only in phase counting mode or when counting overflow/underflow on another channel. In other cases they function as up-counters.

The TCNT registers are 16-bit counters. The TPU has six TCNT counters, one for each channel.

The TCNT counters are initialized to H'0000 by a reset, and in hardware standby mode.

The TCNT counters cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

### 10.2.7 Timer General Register (TGR)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TGR registers are 16-bit registers with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers*. The TGR registers are initialized to H'FFFF by a reset, and in hardware standby mode.

The TGR registers cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

Note: * TGR buffer register combinations are TGRA—TGRC and TGRB—TGRD.

### 10.2.8 Timer Start Register (TSTR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

TSTR is an 8-bit readable/writable register that selects operation/stoppage for channels 0 to 5. TSTR is initialized to H'00 by a reset, and in hardware standby mode.

When setting the operating mode in TMDR or the TCNT count clock in TCR, TCNT counter operation should first be stopped.

**Bits 7 and 6—Reserved:** Should always be written with 0.

**Bits 5 to 0—Counter Start 5 to 0 (CST5 to CST0):** These bits select operation or stoppage for TCNT.

**Bit n**

| CSTn | Description | |
|---|---|---|
| 0 | TCNTn count operation is stopped | (Initial value) |
| 1 | TCNTn performs count operation | |

n = 5 to 0

Note: If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.

### 10.2.9 Timer Synchro Register (TSYR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

TSYR is an 8-bit readable/writable register that selects independent operation or synchronous operation for the channel 0 to 5 TCNT counters. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

TSYR is initialized to H'00 by a reset, and in hardware standby mode.

**Bits 7 and 6—Reserved:** Should always be written with 0..

**Bits 5 to 0—Timer Synchro 5 to 0 (SYNC5 to SYNC0):** These bits select whether operation is independent of or synchronized with other channels.

When synchronous operation is selected, synchronous presetting of multiple channels*[1], and synchronous clearing through counter clearing on another channel*[2] are possible.

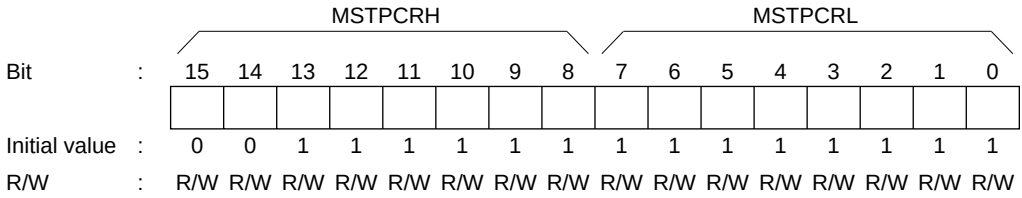
**Bit n**

| SYNCn | Description |
|---|---|
| 0 | TCNTn operates independently (TCNT presetting/clearing is unrelated to other channels) (Initial value) |
| 1 | TCNTn performs synchronous operation<br>TCNT synchronous presetting/synchronous clearing is possible |

n = 5 to 0

Notes: 1. To set synchronous operation, the SYNC bits for at least two channels must be set to 1.
2. To set synchronous clearing, in addition to the SYNC bit , the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.

### 10.2.10 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP13 bit in MSTPCR is set to 1, TPU operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 13—Module Stop (MSTP13):** Specifies the TPU module stop mode.

**Bit 13**

| MSTP13 | Description | |
|---|---|---|
| 0 | TPU module stop mode cleared | |
| 1 | TPU module stop mode set | (Initial value) |

## 10.3 Interface to Bus Master

### 10.3.1 16-Bit Registers

TCNT and TGR are 16-bit registers. As the data bus to the bus master is 16 bits wide, these registers can be read and written to in 16-bit units.

These registers cannot be read or written to in 8-bit units; 16-bit access must always be used.

An example of 16-bit register access operation is shown in figure 10-2.

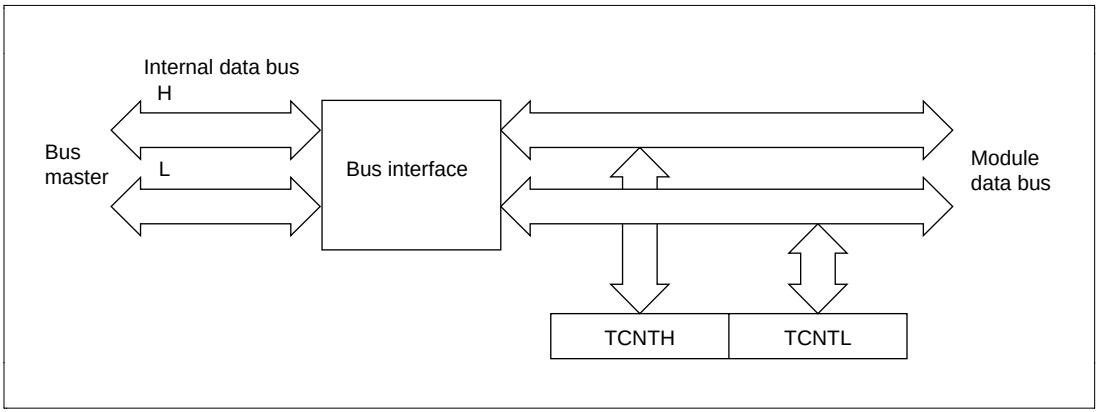



**Figure 10-2 16-Bit Register Access Operation [Bus Master ↔ TCNT (16 Bits)]**

### 10.3.2 8-Bit Registers

Registers other than TCNT and TGR are 8-bit. As the data bus to the CPU is 16 bits wide, these registers can be read and written to in 16-bit units. They can also be read and written to in 8-bit units.

Examples of 8-bit register access operation are shown in figures 10-3, 10-4, and 10-5.

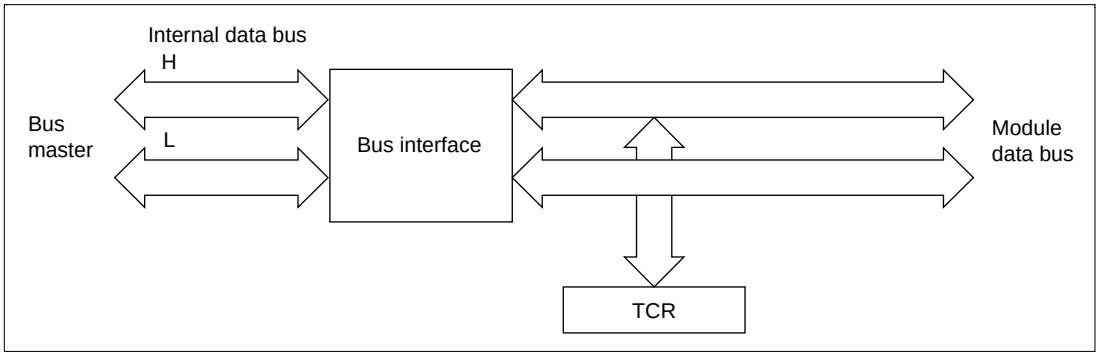



**Figure 10-3 8-Bit Register Access Operation [Bus Master ↔ TCR (Upper 8 Bits)]**

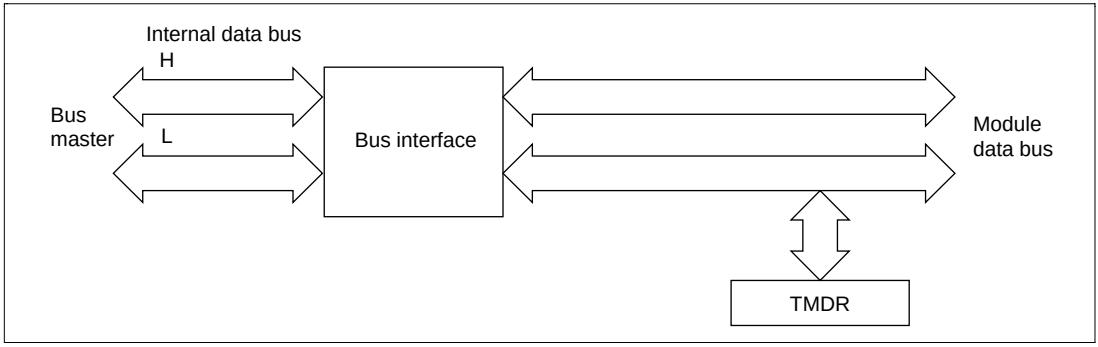



**Figure 10-4 8-Bit Register Access Operation [Bus Master ↔ TMDR (Lower 8 Bits)]**

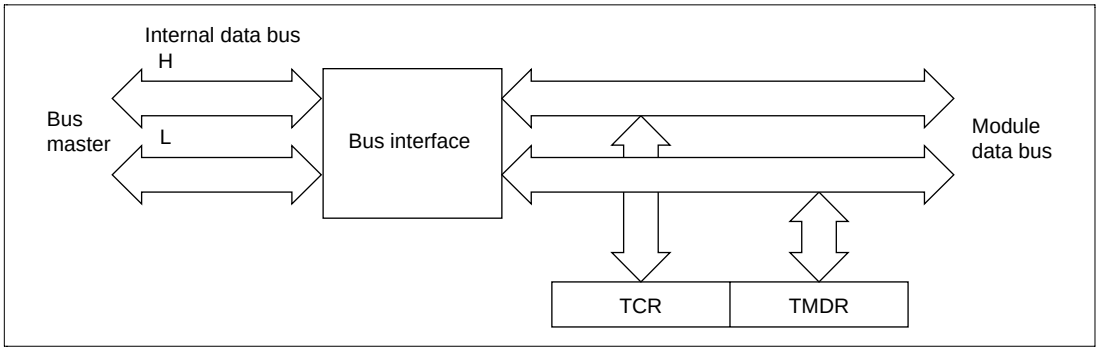



**Figure 10-5 8-Bit Register Access Operation [Bus Master ↔ TCR and TMDR (16 Bits)]**

## 10.4 Operation

### 10.4.1 Overview

Operation in each mode is outlined below.

**Normal Operation:** Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, synchronous counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

**Synchronous Operation:** When synchronous operation is designated for a channel, TCNT for that channel performs synchronous presetting. That is, when TCNT for a channel designated for synchronous operation is rewritten, the TCNT counters for the other channels are also rewritten at the same time. Synchronous clearing of the TCNT counters is also possible by setting the timer synchronization bits in TSYR for channels designated for synchronous operation.

**Buffer Operation**

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the relevant channel is transferred to TGR.

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transfer to TGR and the value previously held in TGR is transferred to the buffer register.

**Cascaded Operation:** The channel 1 counter (TCNT1), channel 2 counter (TCNT2), channel 4 counter (TCNT4), and channel 5 counter (TCNT5) can be connected together to operate as a 32-bit counter.

**PWM Mode:** In this mode, a PWM waveform is output. The output level can be set by means of TIOR. A PWM waveform with a duty of between 0% and 100% can be output, according to the setting of each TGR register.

**Phase Counting Mode:** In this mode, TCNT is incremented or decremented by detecting the phases of two clocks input from the external clock input pins in channels 1, 2, 4, and 5. When phase counting mode is set, the corresponding TCLK pin functions as the clock pin, and TCNT performs up- or down-counting.

This can be used for two-phase encoder pulse input.

### 10.4.2 Basic Functions

**Counter Operation:** When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

- Example of count operation setting procedure

  Figure 10-6 shows an example of the count operation setting procedure.

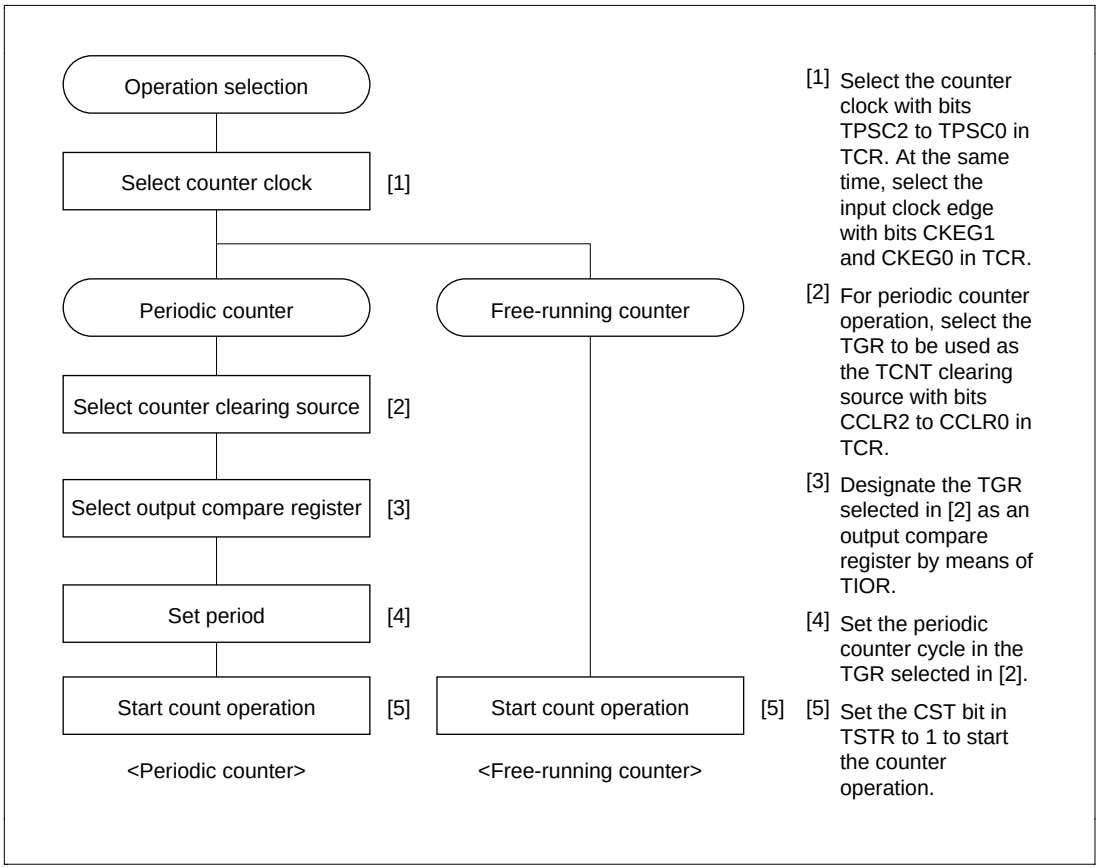



**Figure 10-6 Example of Counter Operation Setting Procedure**

- Free-running count operation and periodic count operation

Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

Figure 10-7 illustrates free-running counter operation.

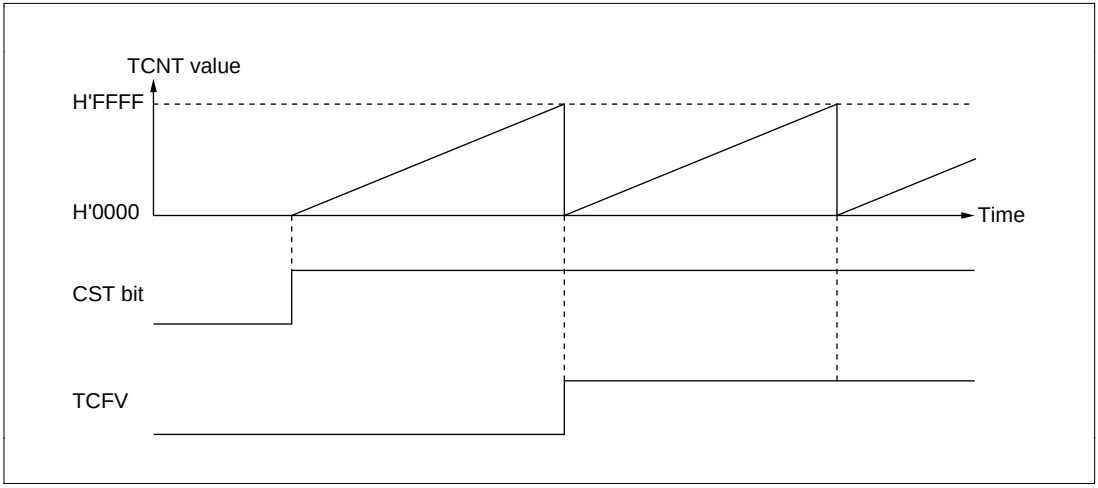



**Figure 10-7 Free-Running Counter Operation**

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts up-count operation as periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 10-8 illustrates periodic counter operation.

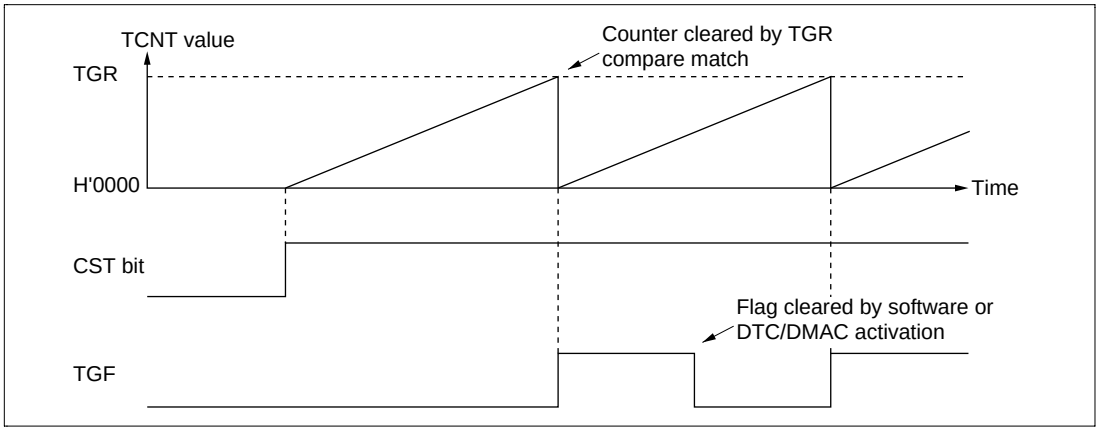



**Figure 10-8 Periodic Counter Operation**

**Waveform Output by Compare Match:** The TPU can perform 0, 1, or toggle output from the corresponding output pin using compare match.

- Example of setting procedure for waveform output by compare match
  Figure 10-9 shows an example of the setting procedure for waveform output by compare match

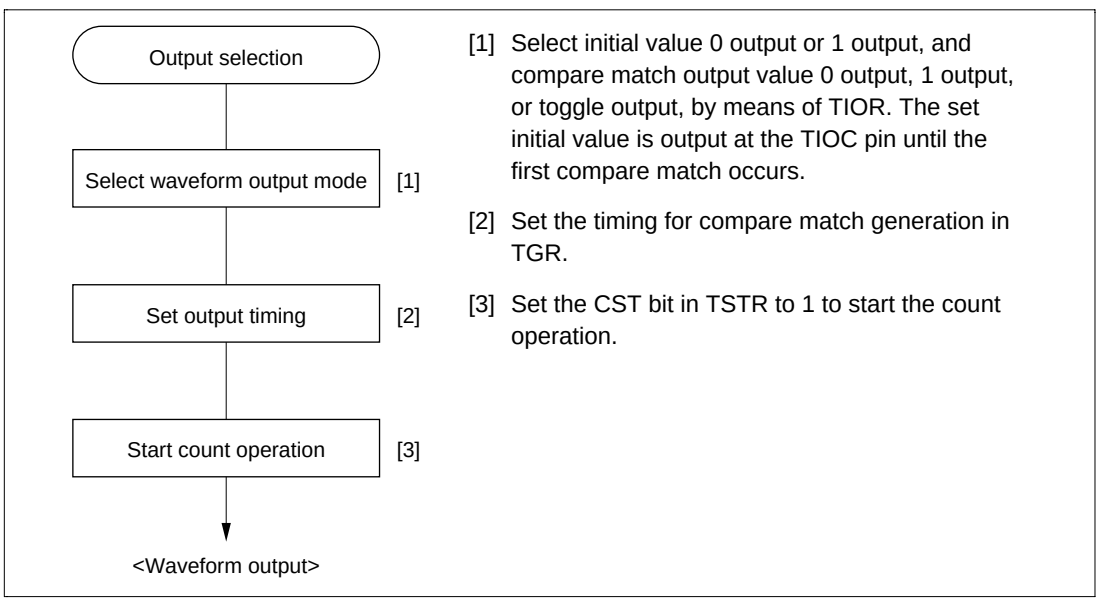



**Figure 10-9 Example Of Setting Procedure For Waveform Output By Compare Match**

- Examples of waveform output operation

  Figure 10-10 shows an example of 0 output/1 output.

  In this example TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level coincide, the pin level does not change.

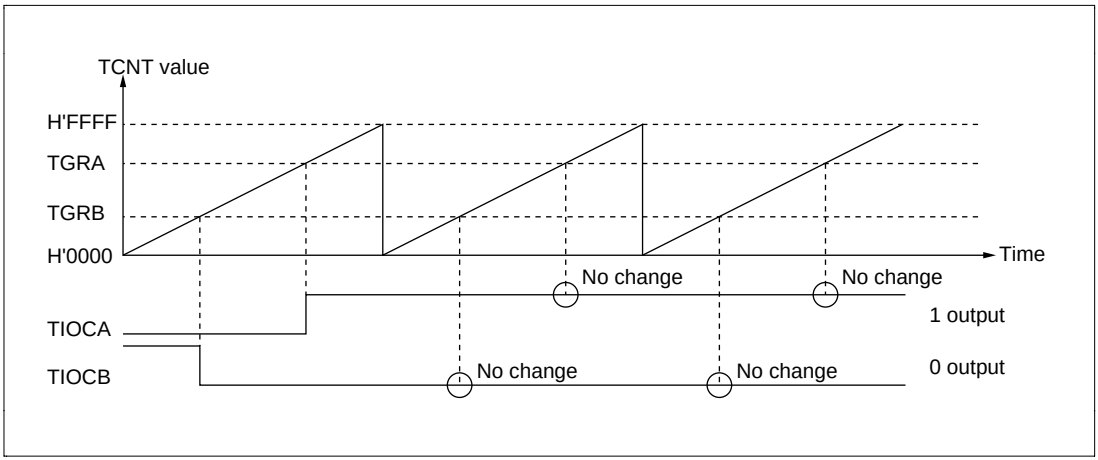



**Figure 10-10 Example of 0 Output/1 Output Operation**

Figure 10-11 shows an example of toggle output.

In this example TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.

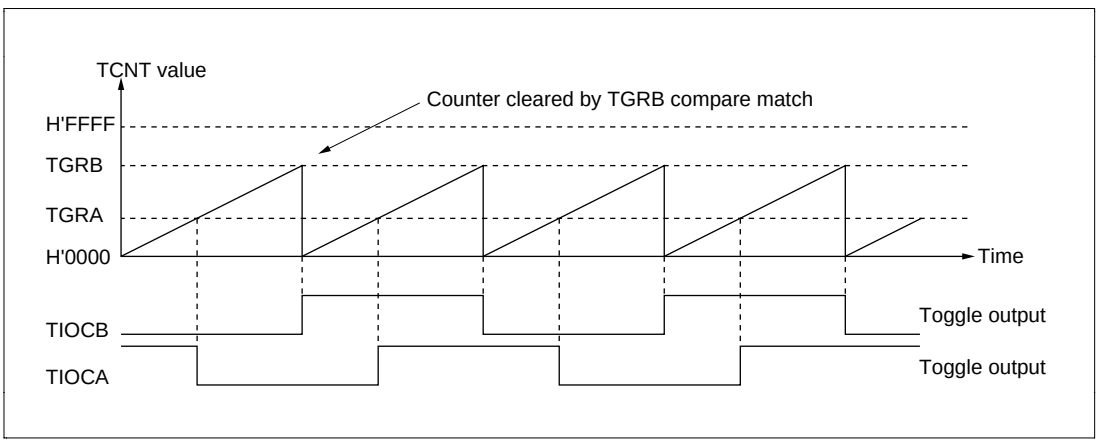



**Figure 10-11 Example of Toggle Output Operation**

**Input Capture Function:** The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detected edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3, ø/1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if ø/1 is selected.

- Example of input capture operation setting procedure
  Figure 10-12 shows an example of the input capture operation setting procedure.

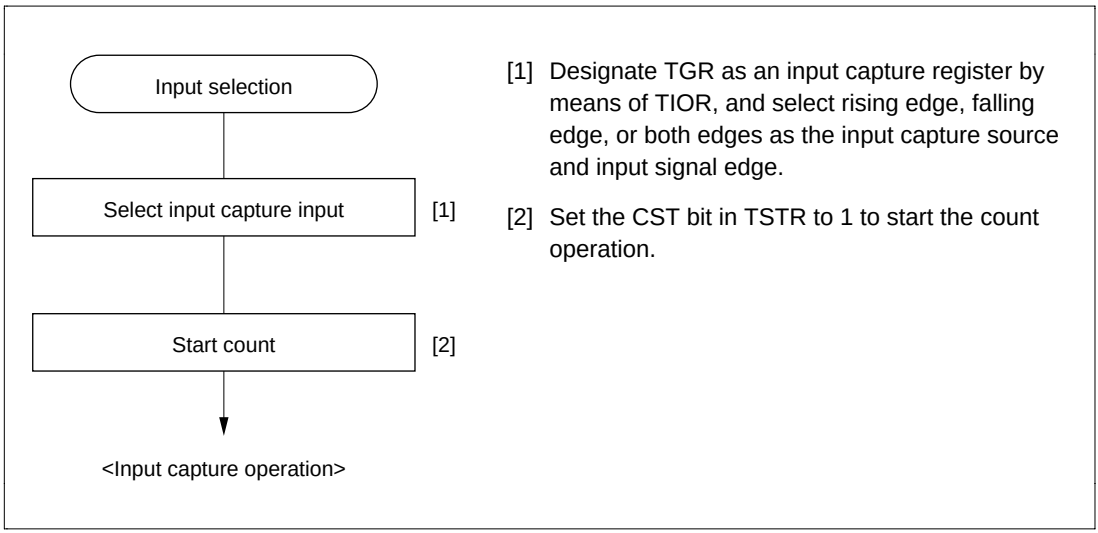



**Figure 10-12 Example of Input Capture Operation Setting Procedure**

- Example of input capture operation

  Figure 10-13 shows an example of input capture operation.

  In this example both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.

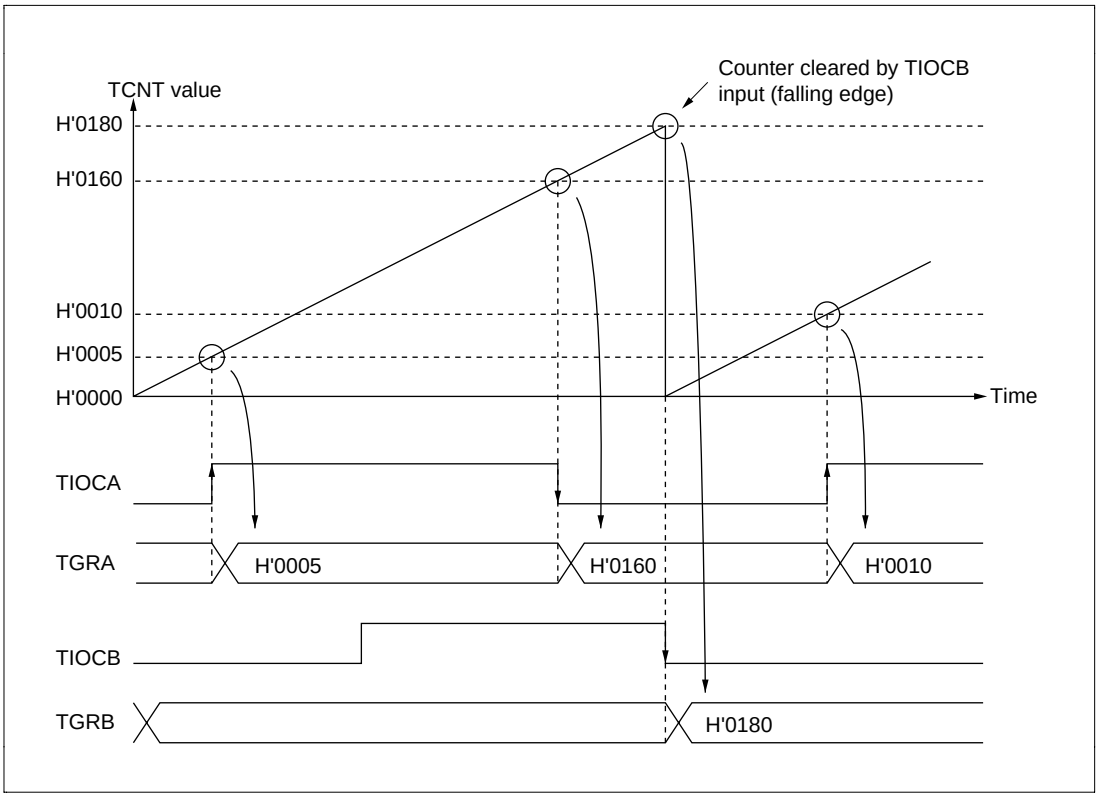



**Figure 10-13 Example of Input Capture Operation**

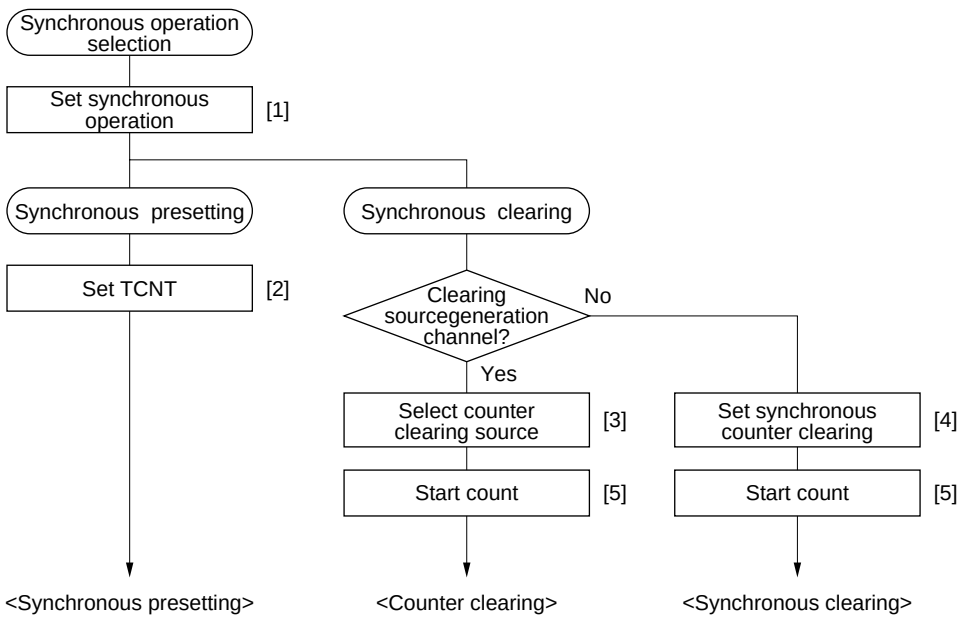
### 10.4.3 Synchronous Operation

In synchronous operation, the values in a number of TCNT counters can be rewritten simultaneously (synchronous presetting). Also, a number of TCNT counters can be cleared simultaneously by making the appropriate setting in TCR (synchronous clearing).

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

**Example of Synchronous Operation Setting Procedure:** Figure 10-14 shows an example of the synchronous operation setting procedure.




[1] Set to 1 the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation.

[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.

[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.

[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.

[5] Set to 1 the CST bits in TSTR for the relevant channels, to start the count operation.

**Figure 10-14 Example of Synchronous Operation Setting Procedure**

**Example of Synchronous Operation:** Figure 10-15 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGR0B compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOC0A, TIOC1A, and TIOC2A. At this time, synchronous presetting, and synchronous clearing by TGR0B compare match, is performed for channel 0 to 2 TCNT counters, and the data set in TGR0B is used as the PWM cycle.

For details of PWM modes, see section 10.4.6, PWM Modes.

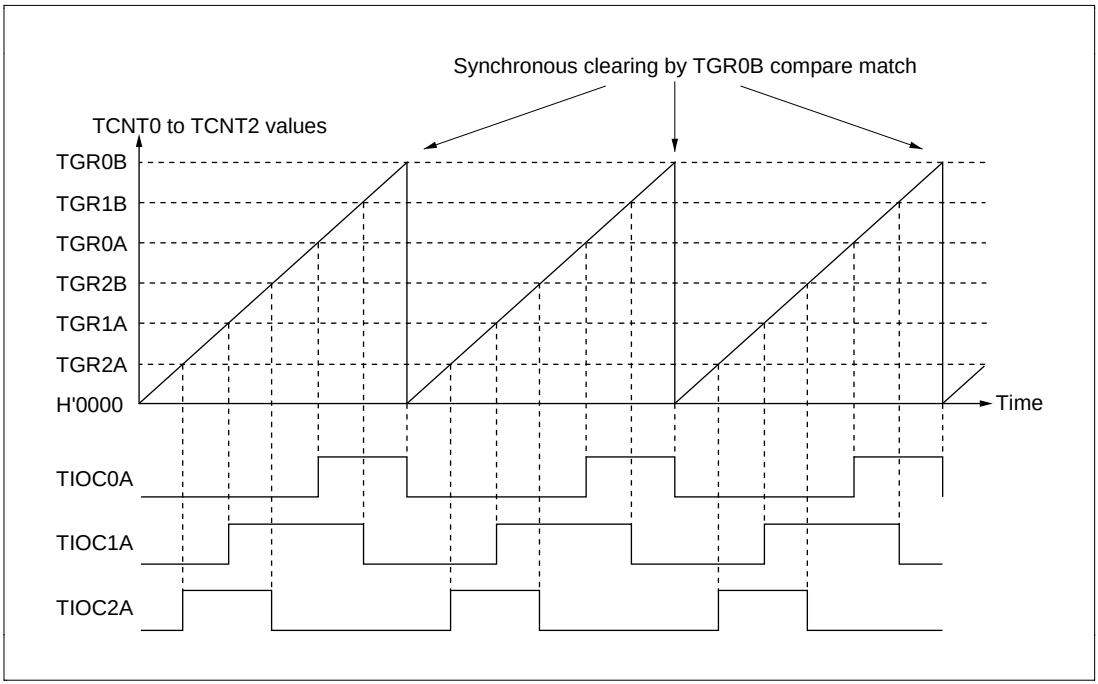



**Figure 10-15 Example of Synchronous Operation**

### 10.4.4 Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or as a compare match register.

Table 10-5 shows the register combinations used in buffer operation.

**Table 10-5 Register Combinations in Buffer Operation**

| Channel | Timer General Register | Buffer Register |
|---|---|---|
| 0 | TGR0A | TGR0C |
| | TGR0B | TGR0D |
| 3 | TGR3A | TGR3C |
| | TGR3B | TGR3D |

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

  This operation is illustrated in figure 10-16.

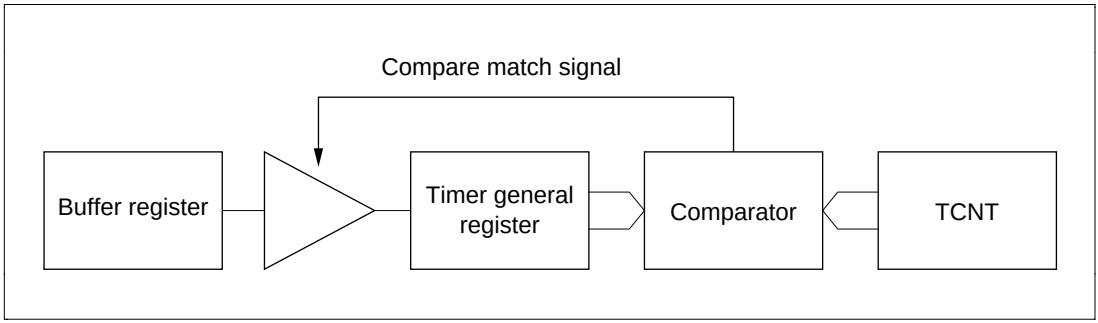



**Figure 10-16 Compare Match Buffer Operation**

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in the timer general register is transferred to the buffer register.

  This operation is illustrated in figure 10-17.

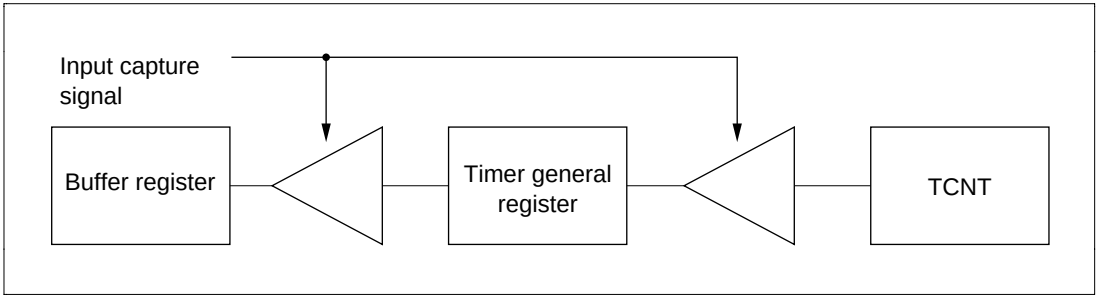



**Figure 10-17 Input Capture Buffer Operation**

**Example of Buffer Operation Setting Procedure:** Figure 10-18 shows an example of the buffer operation setting procedure.

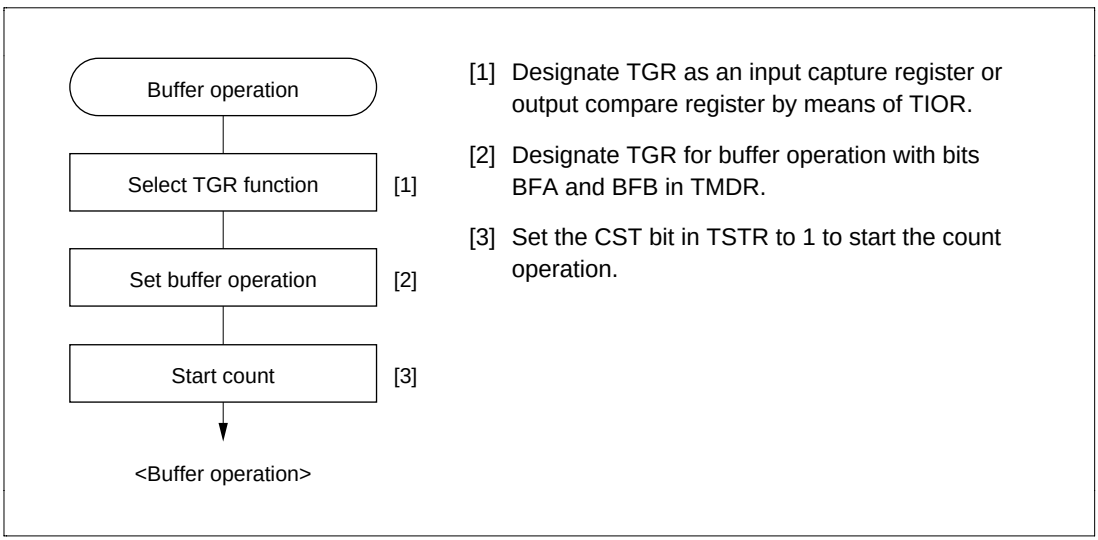



**Figure 10-18 Example of Buffer Operation Setting Procedure**

**Examples of Buffer Operation**

- When TGR is an output compare register

  Figure 10-19 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

  As buffer operation has been set, when compare match A occurs the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

  For details of PWM modes, see section 10.4.6, PWM Modes.

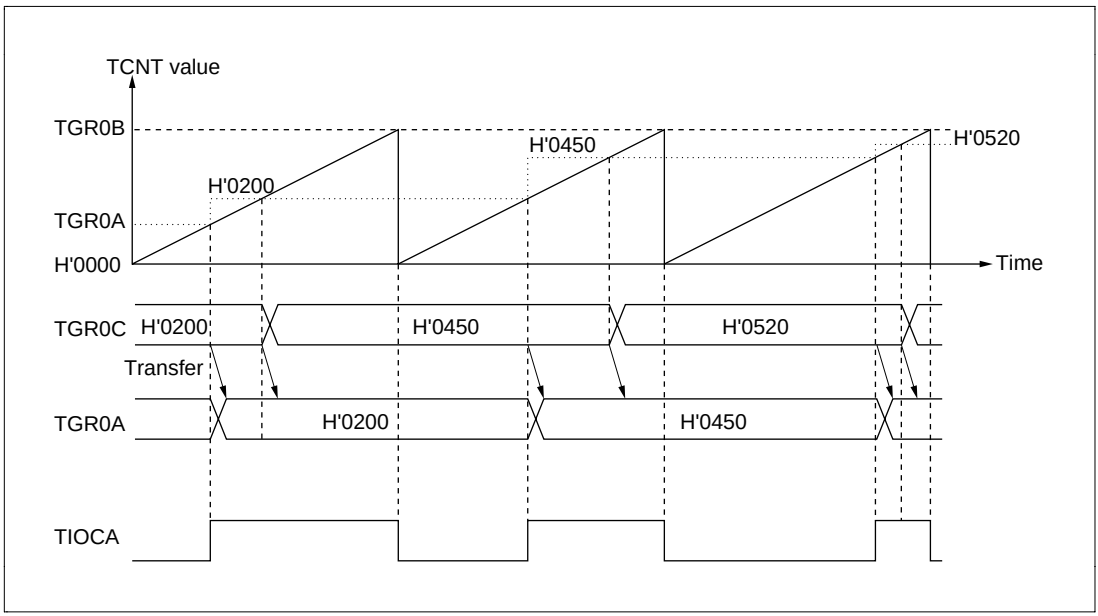



**Figure 10-19 Example of Buffer Operation (1)**

- When TGR is an input capture register

  Figure 10-20 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

  Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

  As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

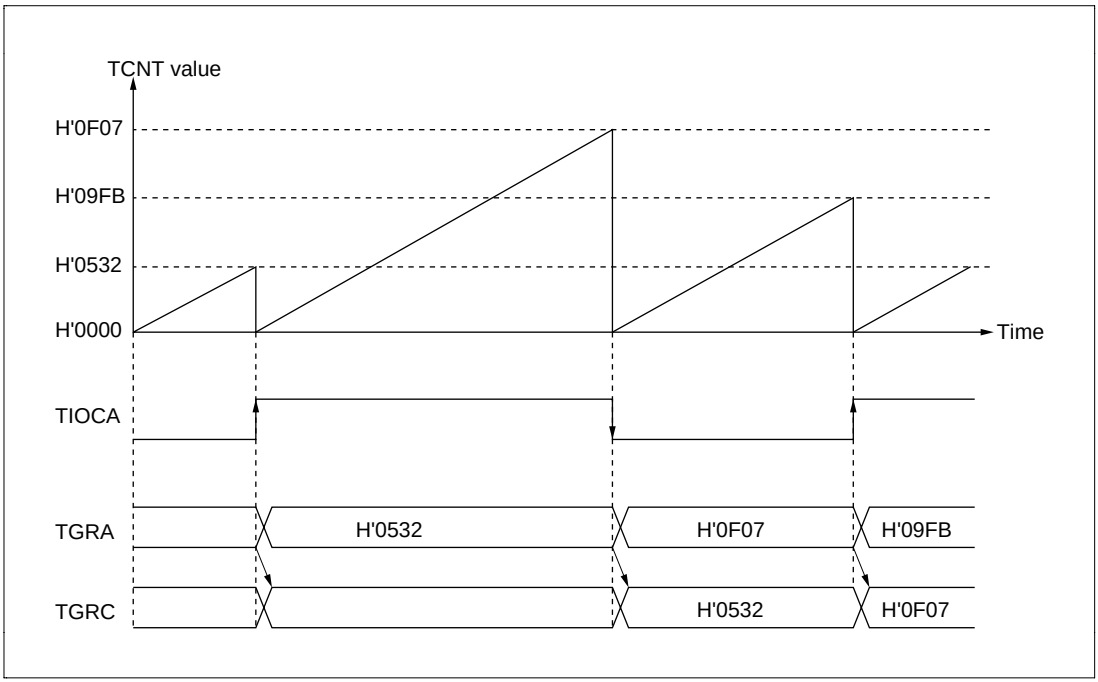



**Figure 10-20 Example of Buffer Operation (2)**

### 10.4.5 Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock upon overflow/underflow of TCNT2 (TCNT5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 10-6 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

**Table 10-6 Cascaded Combinations**

| Combination | Upper 16 Bits | Lower 16 Bits |
|---|---|---|
| Channels 1 and 2 | TCNT1 | TCNT2 |
| Channels 4 and 5 | TCNT4 | TCNT5 |

**Example of Cascaded Operation Setting Procedure:** Figure 10-21 shows an example of the setting procedure for cascaded operation.

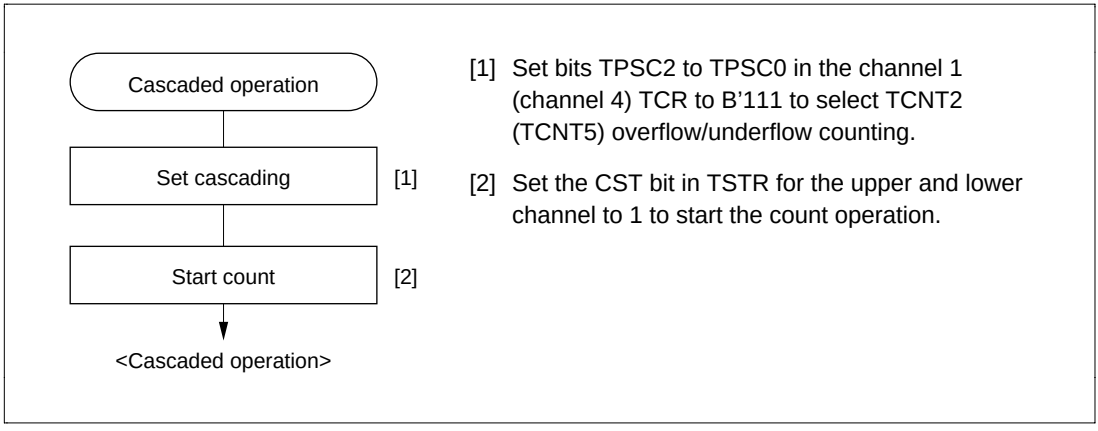



**Figure 10-21 Cascaded Operation Setting Procedure**

**Examples of Cascaded Operation:** Figure 10-22 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, TGR1A and TGR2A have been designated as input capture registers, and TIOC pin rising edge has been selected.

When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGR1A, and the lower 16 bits to TGR2A.

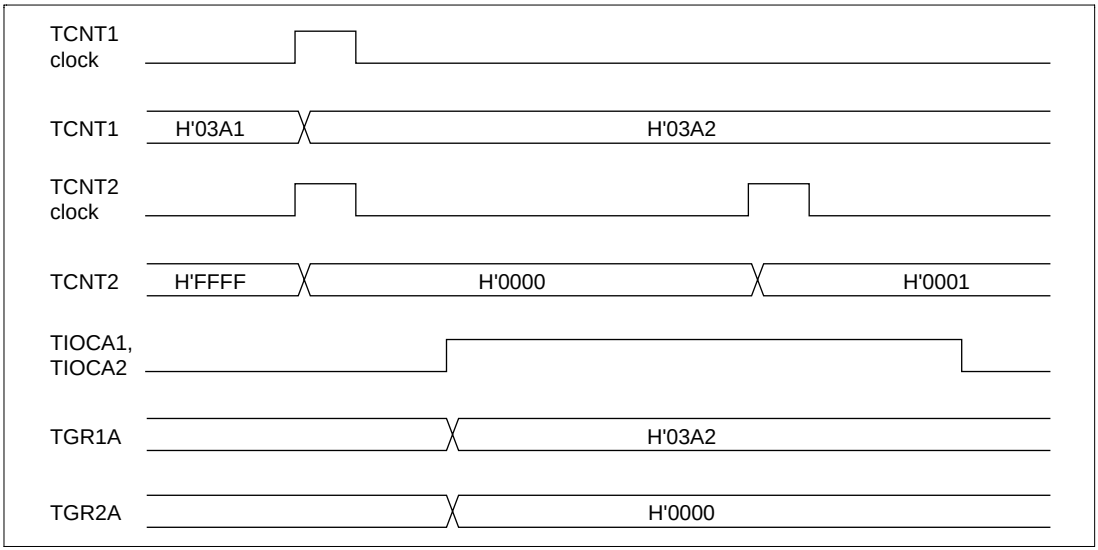



**Figure 10-22 Example of Cascaded Operation (1)**

Figure 10-23 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, and phase counting mode has been designated for channel 2.

TCNT1 is incremented by TCNT2 overflow and decremented by TCNT2 underflow.

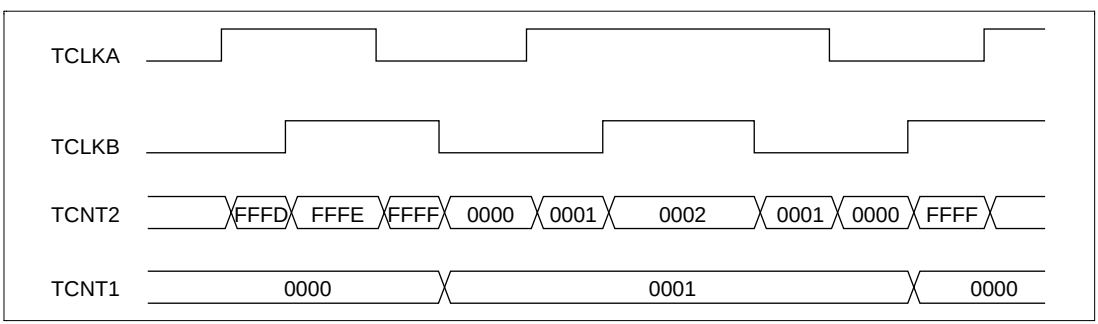



**Figure 10-23 Example of Cascaded Operation (2)**

### 10.4.6 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0, 1, or toggle output can be selected as the output level in response to compare match of each TGR.

Designating TGR compare match as the counter clearing source enables the period to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

- PWM mode 1

  PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The output specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR is output from the TIOCA and TIOCC pins at compare matches A and C, and the output specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR is output at compare matches B and D. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

  In PWM mode 1, a maximum 8-phase PWM output is possible.

- PWM mode 2

  PWM output is generated using one TGR as the cycle register and the others as duty registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronization register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty registers are identical, the output value does not change when a compare match occurs.

  In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

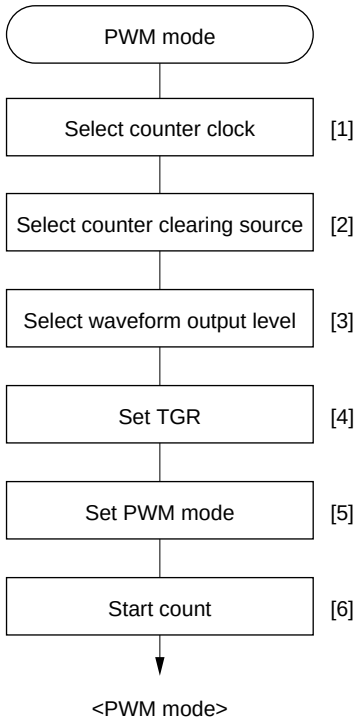
The correspondence between PWM output pins and registers is shown in table 10-7.

**Table 10-7 PWM Output Registers and Output Pins**

| | | Output Pins | |
|---|---|---|---|
| **Channel** | **Registers** | **PWM Mode 1** | **PWM Mode 2** |
| 0 | TGR0A | TIOCA0 | TIOCA0 |
| | TGR0B | | TIOCB0 |
| | TGR0C | TIOCC0 | TIOCC0 |
| | TGR0D | | TIOCD0 |
| 1 | TGR1A | TIOCA1 | TIOCA1 |
| | TGR1B | | TIOCB1 |
| 2 | TGR2A | TIOCA2 | TIOCA2 |
| | TGR2B | | TIOCB2 |
| 3 | TGR3A | TIOCA3 | TIOCA3 |
| | TGR3B | | TIOCB3 |
| | TGR3C | TIOCC3 | TIOCC3 |
| | TGR3D | | TIOCD3 |
| 4 | TGR4A | TIOCA4 | TIOCA4 |
| | TGR4B | | TIOCB4 |
| 5 | TGR5A | TIOCA5 | TIOCA5 |
| | TGR5B | | TIOCB5 |

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the period is set.

**Example of PWM Mode Setting Procedure:** Figure 10-24 shows an example of the PWM mode setting procedure.




[1] Select the counter clock with bits TPSC2 to TPSC0 in TCR. At the same time, select the input clock edge with bits CKEG1 and CKEG0 in TCR.

[2] Use bits CCLR2 to CCLR0 in TCR to select the TGR to be used as the TCNT clearing source.

[3] Use TIOR to designate the TGR as an output compare register, and select the initial value and output value.

[4] Set the cycle in the TGR selected in [2], and set the duty in the other the TGR.

[5] Select the PWM mode with bits MD3 to MD0 in TMDR.

[6] Set the CST bit in TSTR to 1 to start the count operation.

**Figure 10-24 Example of PWM Mode Setting Procedure**

**Examples of PWM Mode Operation:** Figure 10-25 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the period, and the values set in TGRB registers as the duty.

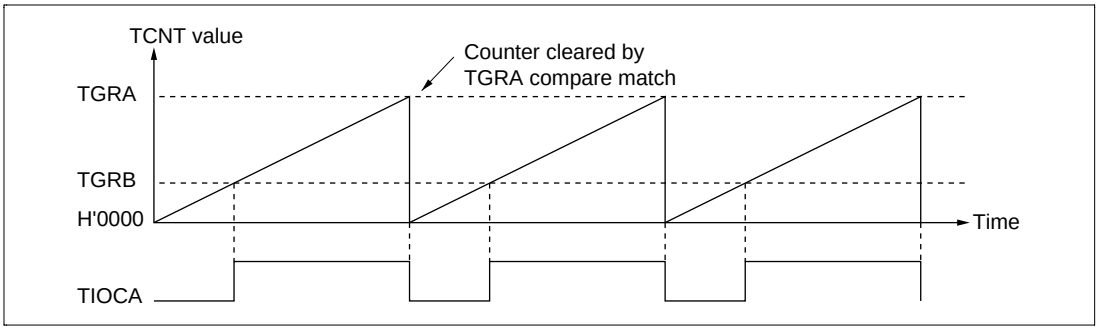



**Figure 10-25 Example of PWM Mode Operation (1)**

Figure 10-26 shows an example of PWM mode 2 operation.

In this example, synchronous operation is designated for channels 0 and 1, TGR1B compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGR0A to TGR0D, TGR1A), to output a 5-phase PWM waveform.

In this case, the value set in TGR1B is used as the cycle, and the values set in the other TGRs as the duty.

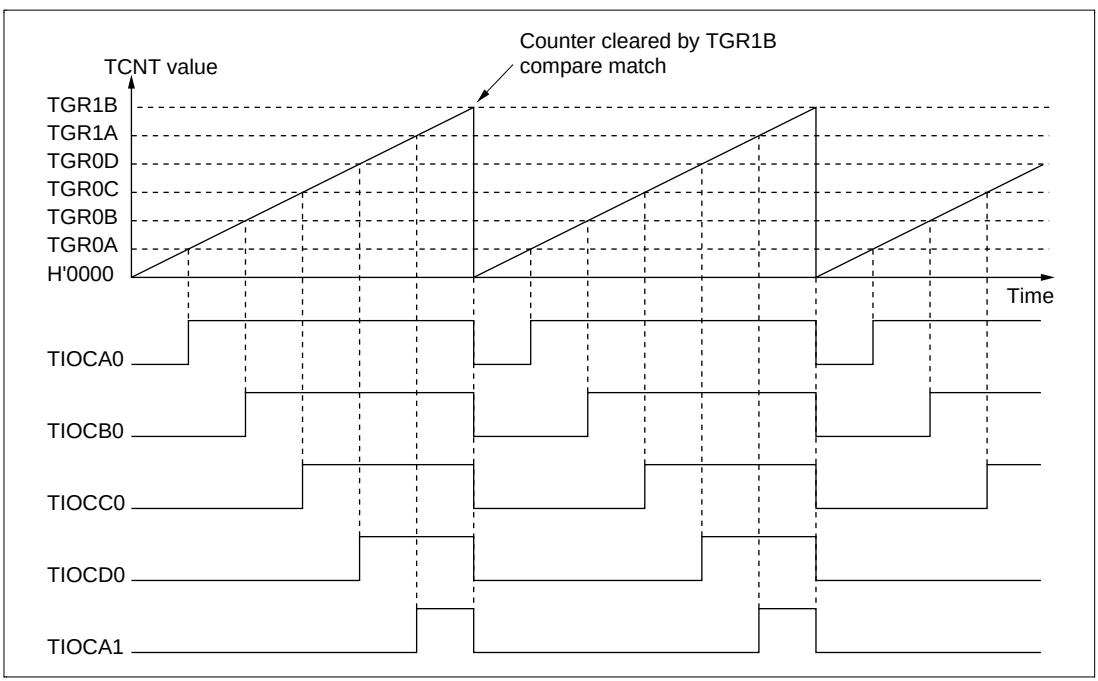



**Figure 10-26 Example of PWM Mode Operation (2)**

Figure 10-27 shows examples of PWM waveform output with 0% duty and 100% duty in PWM mode.

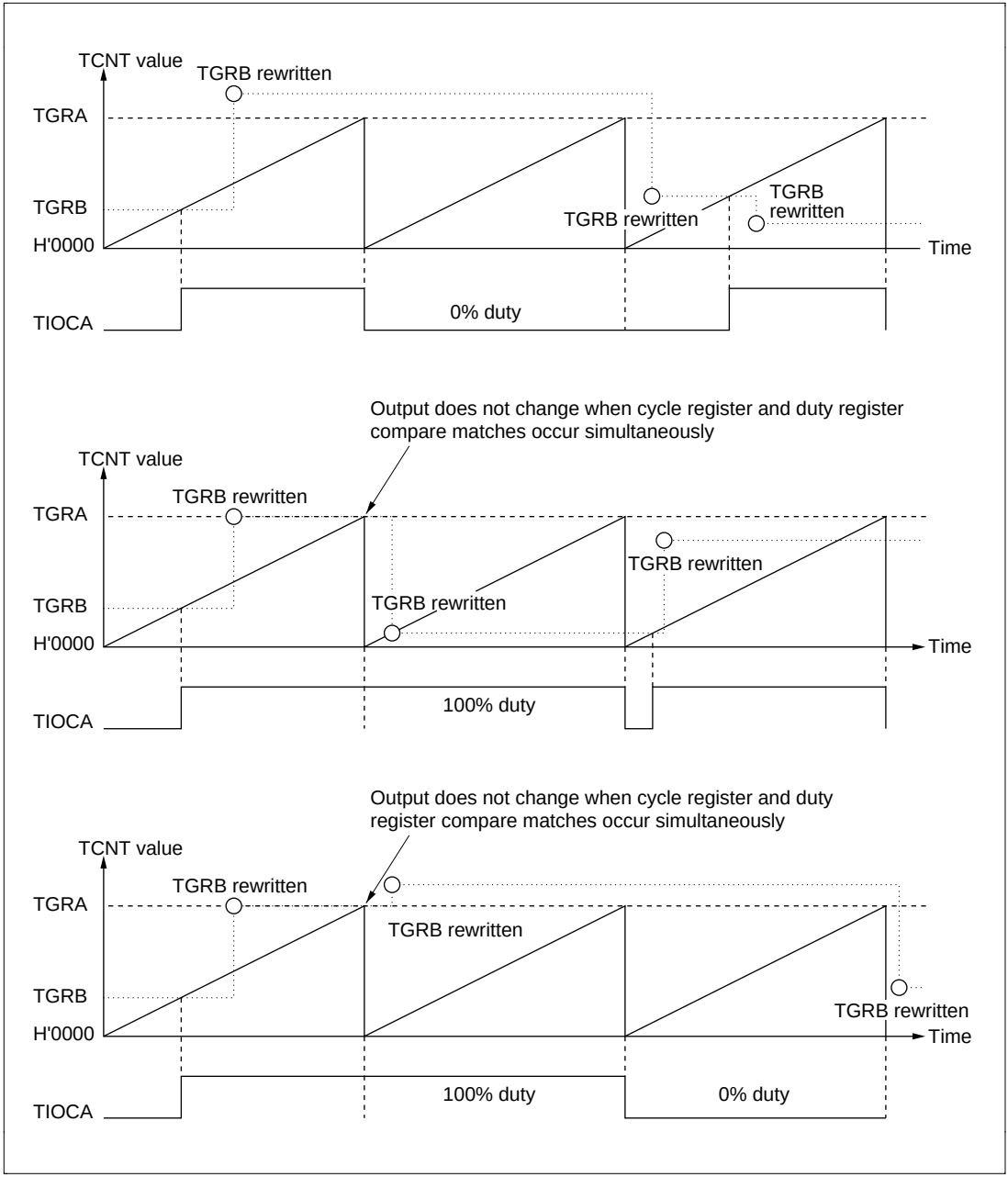



**Figure 10-27 Example of PWM Mode Operation (3)**

### 10.4.7 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 10-8 shows the correspondence between external clock pins and channels.

**Table 10-8 Phase Counting Mode Clock Input Pins**

| | External Clock Pins | |
|---|---|---|
| **Channels** | **A-Phase** | **B-Phase** |
| When channel 1 or 5 is set to phase counting mode | TCLKA | TCLKB |
| When channel 2 or 4 is set to phase counting mode | TCLKC | TCLKD |

**Example of Phase Counting Mode Setting Procedure:** Figure 10-28 shows an example of the phase counting mode setting procedure.

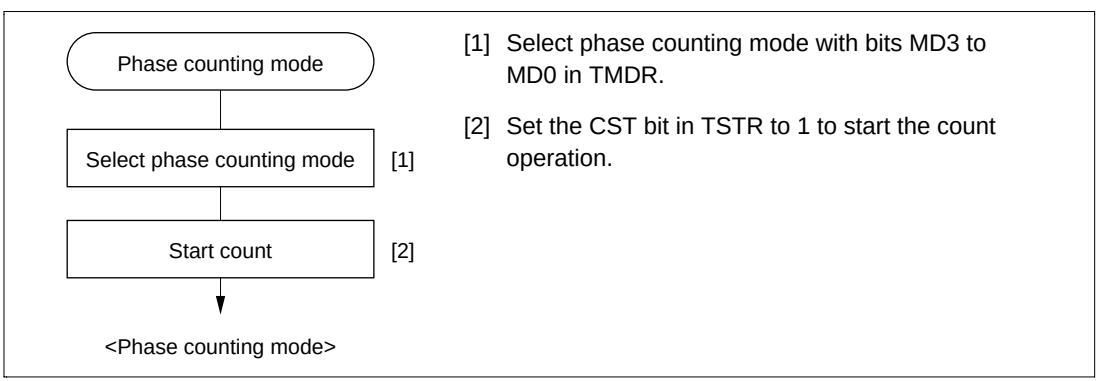



**Figure 10-28 Example of Phase Counting Mode Setting Procedure**

**Examples of Phase Counting Mode Operation:** In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

- Phase counting mode 1

  Figure 10-29 shows an example of phase counting mode 1 operation, and table 10-9 summarizes the TCNT up/down-count conditions.

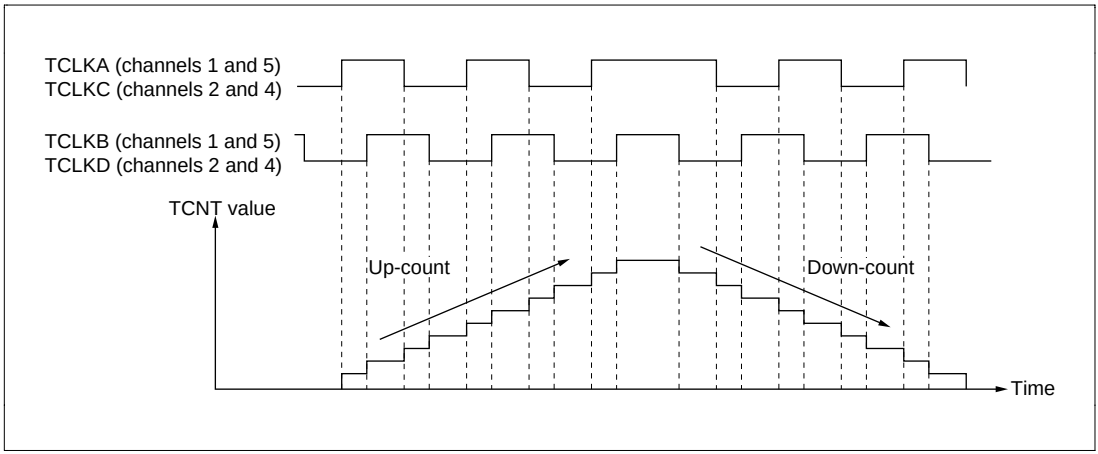



**Figure 10-29 Example of Phase Counting Mode 1 Operation**

**Table 10-9 Up/Down-Count Conditions in Phase Counting Mode 1**

| TCLKA (Channels 1 and 5)<br>TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5)<br>TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | |
| ↓ | Low level | |

Legend

↑ : Rising edge

↓ : Falling edge

- Phase counting mode 2

  Figure 10-30 shows an example of phase counting mode 2 operation, and table 10-10 summarizes the TCNT up/down-count conditions.

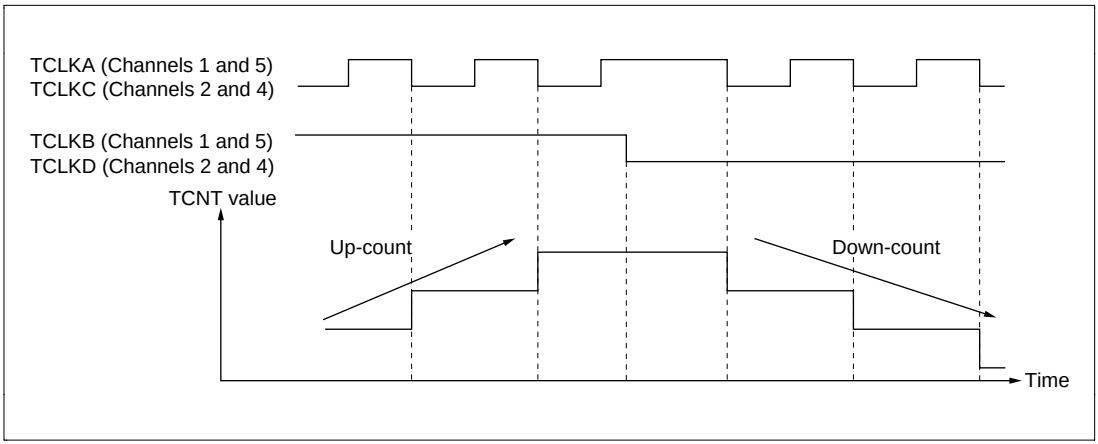



**Figure 10-30 Example of Phase Counting Mode 2 Operation**

**Table 10-10 Up/Down-Count Conditions in Phase Counting Mode 2**

| TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Don't care |
| Low level | ↓ | Don't care |
| ↑ | Low level | Don't care |
| ↓ | High level | Up-count |
| High level | ↓ | Don't care |
| Low level | ↑ | Don't care |
| ↑ | High level | Don't care |
| ↓ | Low level | Down-count |

Legend

↑ : Rising edge

↓ : Falling edge

- Phase counting mode 3

  Figure 10-31 shows an example of phase counting mode 3 operation, and table 10-11 summarizes the TCNT up/down-count conditions.

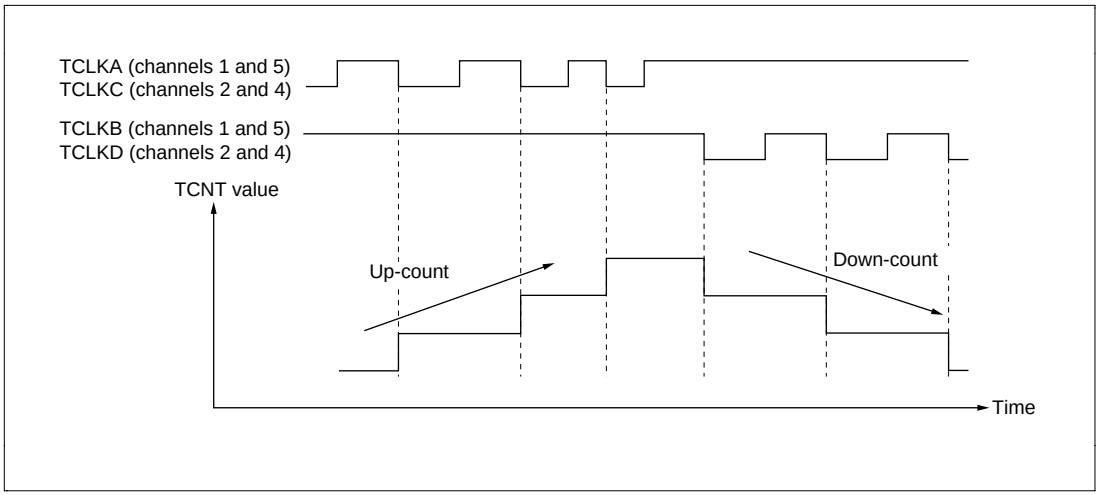



**Figure 10-31 Example of Phase Counting Mode 3 Operation**

**Table 10-11 Up/Down-Count Conditions in Phase Counting Mode 3**

| TCLKA (Channels 1 and 5)<br>TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5)<br>TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ⌐↑ | Don't care |
| Low level | ⌐↓ | Don't care |
| ⌐↑ | Low level | Don't care |
| ⌐↓ | High level | Up-count |
| High level | ⌐↓ | Down-count |
| Low level | ⌐↑ | Don't care |
| ⌐↑ | High level | Don't care |
| ⌐↓ | Low level | Don't care |

Legend

⌐↑ : Rising edge

⌐↓ : Falling edge

- Phase counting mode 4

  Figure 10-32 shows an example of phase counting mode 4 operation, and table 10-12 summarizes the TCNT up/down-count conditions.

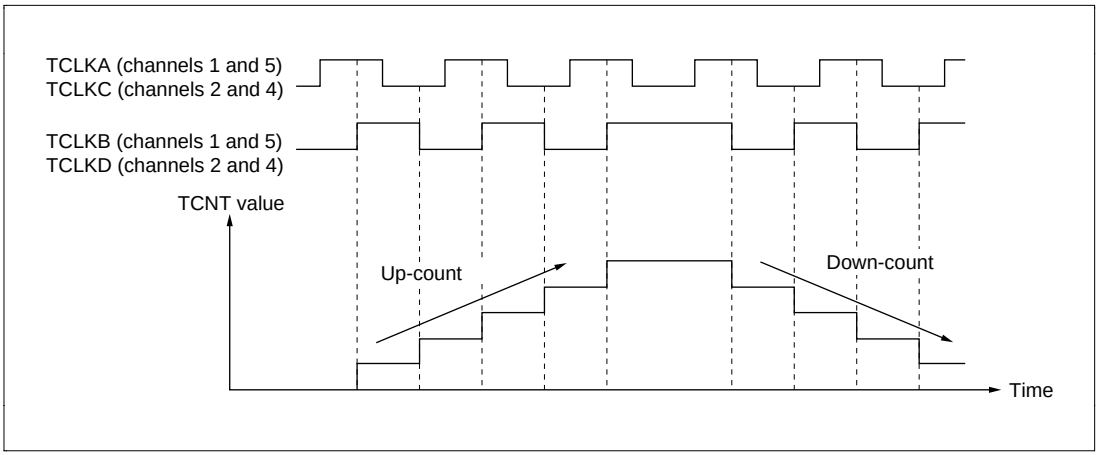



**Figure 10-32 Example of Phase Counting Mode 4 Operation**

**Table 10-12 Up/Down-Count Conditions in Phase Counting Mode 4**

| TCLKA (Channels 1 and 5)<br>TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5)<br>TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | Don't care |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | Don't care |
| ↓ | Low level | |

Legend

↑ : Rising edge

↓ : Falling edge

**Phase Counting Mode Application Example:** Figure 10-33 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGR0C compare match; TGR0A and TGR0C are used for the compare match function, and are set with the speed control period and position control period. TGR0B is used for input capture, with TGR0B and TGR0D operating in buffer mode. The channel 1 counter input clock is designated as the TGR0B input capture source, and detection of the pulse width of 2-phase encoder 4-multiplication pulses is performed.

TGR1A and TGR1B for channel 1 are designated for input capture, channel 0 TGR0A and TGR0C compare matches are selected as the input capture source, and store the up/down-counter values for the control periods.

This procedure enables accurate position/speed detection to be achieved.

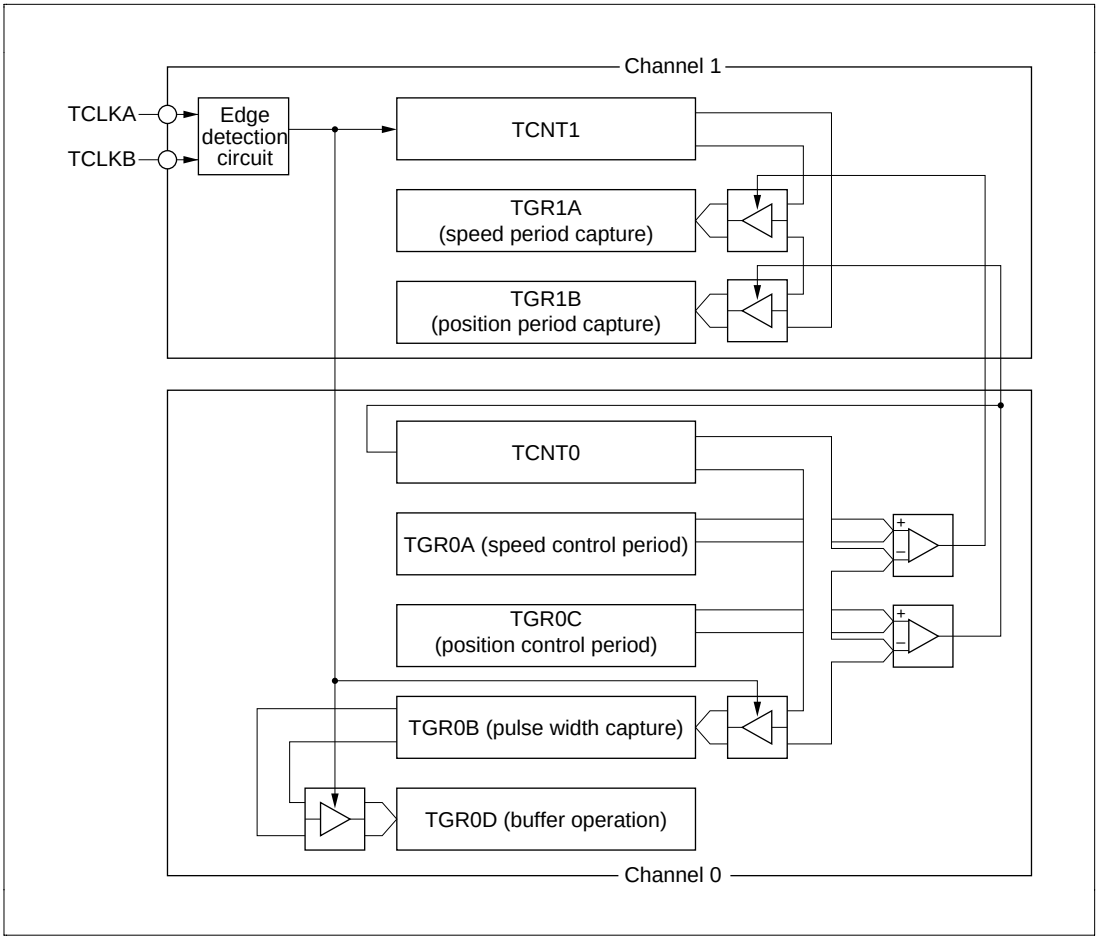



**Figure 10-33 Phase Counting Mode Application Example**

## 10.5 Interrupts

### 10.5.1 Interrupt Sources and Priorities

There are three kinds of TPU interrupt source: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disabled bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priorities can be changed by the interrupt controller, but the priority order within a channel is fixed. For details, see section 5, Interrupt Controller.

Table 10-13 lists the TPU interrupt sources.

**Table 10-13 TPU Interrupts**

| Channel | Interrupt Source | Description | DMAC Activation | DTC Activation | Priority |
|---|---|---|---|---|---|
| 0 | TGI0A | TGR0A input capture/compare match | Possible | Possible | High |
| | TGI0B | TGR0B input capture/compare match | Not possible | Possible | |
| | TGI0C | TGR0C input capture/compare match | Not possible | Possible | |
| | TGI0D | TGR0D input capture/compare match | Not possible | Possible | |
| | TCI0V | TCNT0 overflow | Not possible | Not possible | |
| 1 | TGI1A | TGR1A input capture/compare match | Possible | Possible | |
| | TGI1B | TGR1B input capture/compare match | Not possible | Possible | |
| | TCI1V | TCNT1 overflow | Not possible | Not possible | |
| | TCI1U | TCNT1 underflow | Not possible | Not possible | |
| 2 | TGI2A | TGR2A input capture/compare match | Possible | Possible | |
| | TGI2B | TGR2B input capture/compare match | Not possible | Possible | |
| | TCI2V | TCNT2 overflow | Not possible | Not possible | |
| | TCI2U | TCNT2 underflow | Not possible | Not possible | |
| 3 | TGI3A | TGR3A input capture/compare match | Possible | Possible | |
| | TGI3B | TGR3B input capture/compare match | Not possible | Possible | |
| | TGI3C | TGR3C input capture/compare match | Not possible | Possible | |
| | TGI3D | TGR3D input capture/compare match | Not possible | Possible | |
| | TCI3V | TCNT3 overflow | Not possible | Not possible | |
| 4 | TGI4A | TGR4A input capture/compare match | Possible | Possible | |
| | TGI4B | TGR4B input capture/compare match | Not possible | Possible | |
| | TCI4V | TCNT4 overflow | Not possible | Not possible | |
| | TCI4U | TCNT4 underflow | Not possible | Not possible | |
| 5 | TGI5A | TGR5A input capture/compare match | Possible | Possible | |
| | TGI5B | TGR5B input capture/compare match | Not possible | Possible | |
| | TCI5V | TCNT5 overflow | Not possible | Not possible | |
| | TCI5U | TCNT5 underflow | Not possible | Not possible | Low |

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

**Input Capture/Compare Match Interrupt:** An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a particular channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

**Overflow Interrupt:** An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

**Underflow Interrupt:** An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four underflow interrupts, one each for channels 1, 2, 4, and 5.

### 10.5.2 DTC/DMAC Activation

**DTC Activation:** The DTC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 8, Data Transfer Controller.

A total of 16 TPU input capture/compare match interrupts can be used as DTC activation sources, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

**DMAC Activation:** The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 7, DMA Controller.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

### 10.5.3 A/D Converter Activation

The A/D converter can be activated by the TGRA input capture/compare match for a channel.

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

## 10.6 Operation Timing

### 10.6.1 Input/Output Timing

**TCNT Count Timing:** Figure 10-34 shows TCNT count timing in internal clock operation, and figure 10-35 shows TCNT count timing in external clock operation.

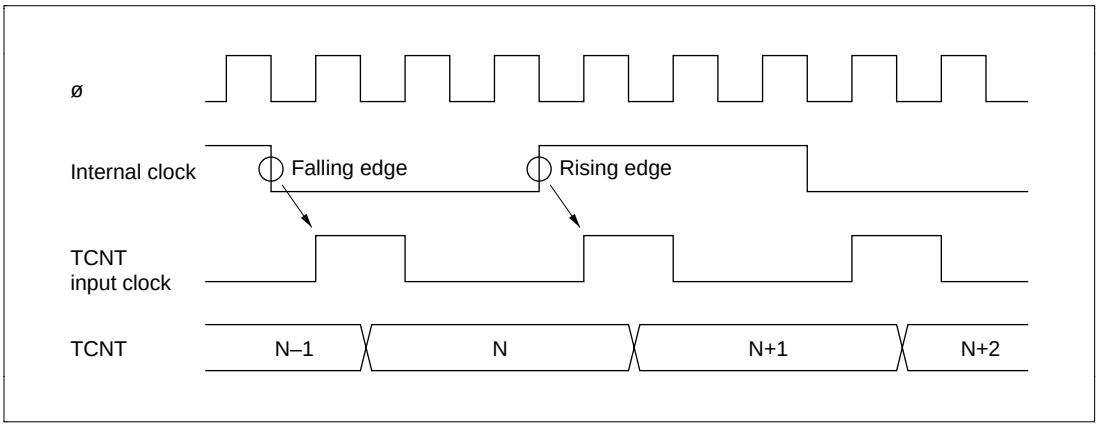



**Figure 10-34 Count Timing in Internal Clock Operation**

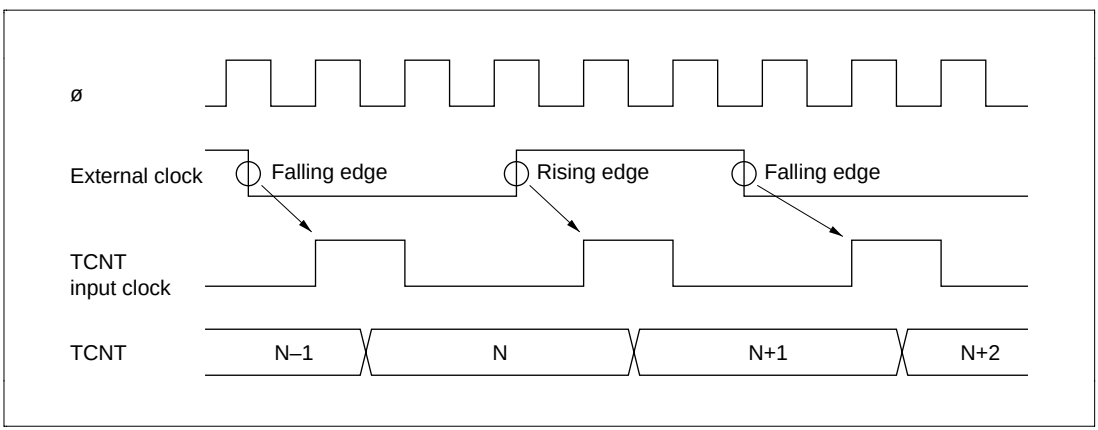



**Figure 10-35 Count Timing in External Clock Operation**

**Output Compare Output Timing:** A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin. After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 10-36 shows output compare output timing.

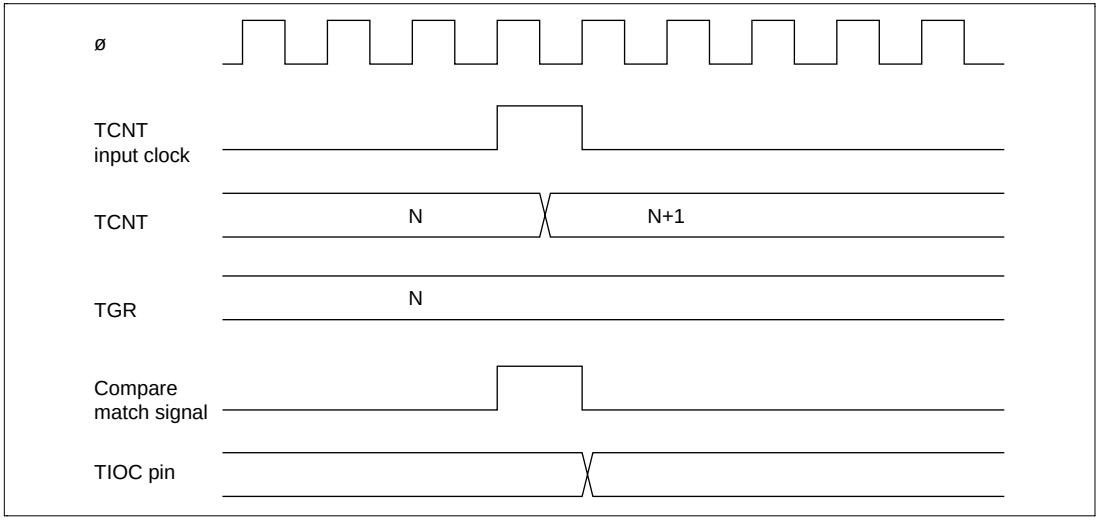



**Figure 10-36 Output Compare Output Timing**

**Input Capture Signal Timing:** Figure 10-37 shows input capture signal timing.

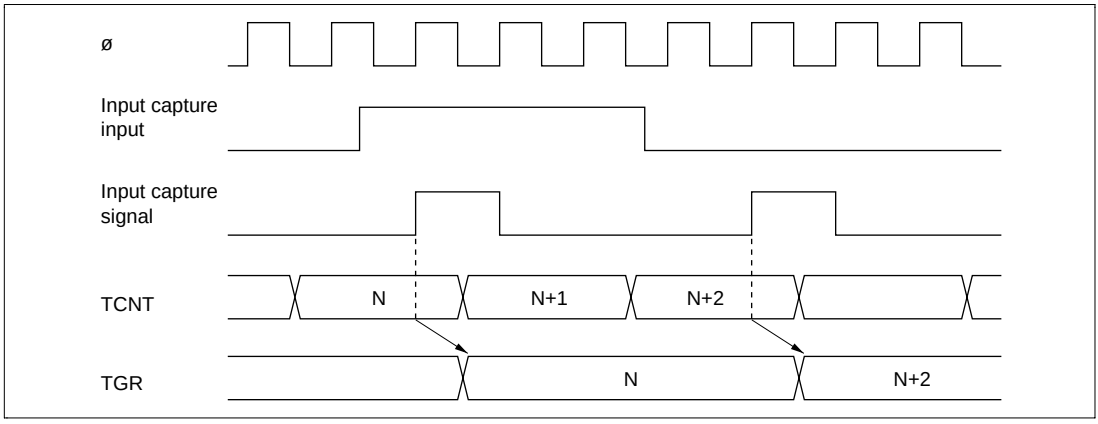



**Figure 10-37 Input Capture Input Signal Timing**

**Timing for Counter Clearing by Compare Match/Input Capture:** Figure 10-38 shows the timing when counter clearing by compare match occurrence is specified, and figure 10-39 shows the timing when counter clearing by input capture occurrence is specified.

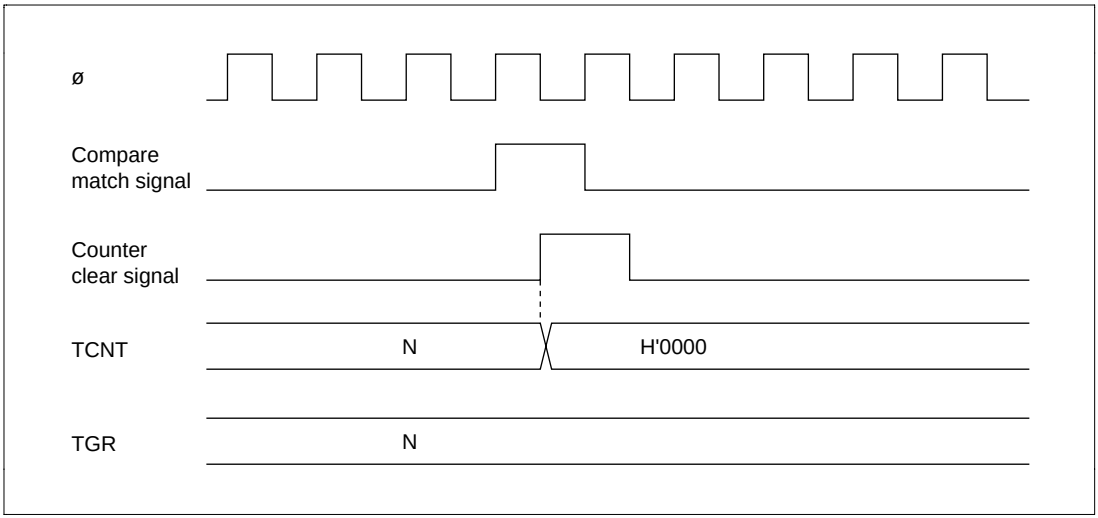



**Figure 10-38 Counter Clear Timing (Compare Match)**

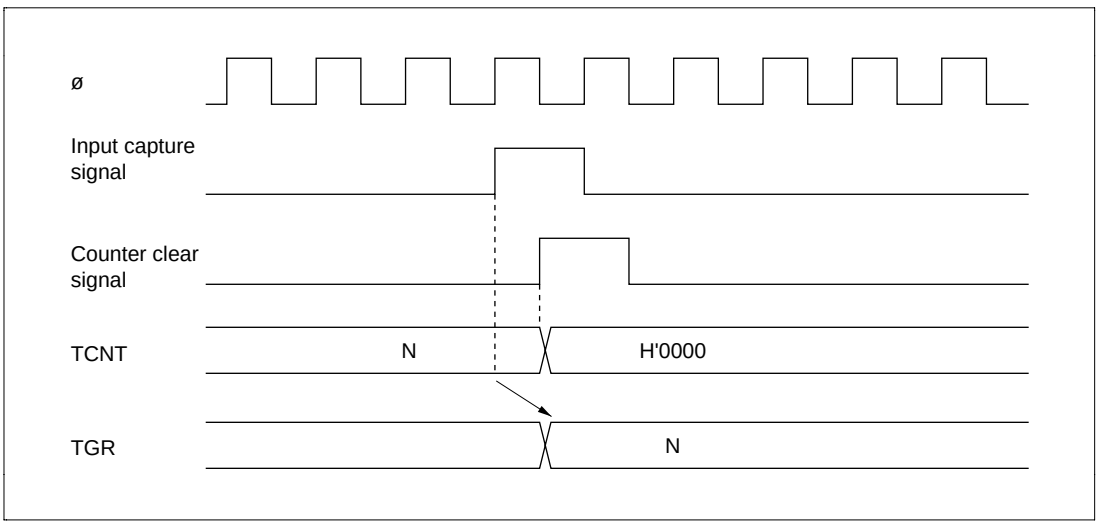



**Figure 10-39 Counter Clear Timing (Input Capture)**

**Buffer Operation Timing:** Figures 10-40 and 10-41 show the timing in buffer operation.

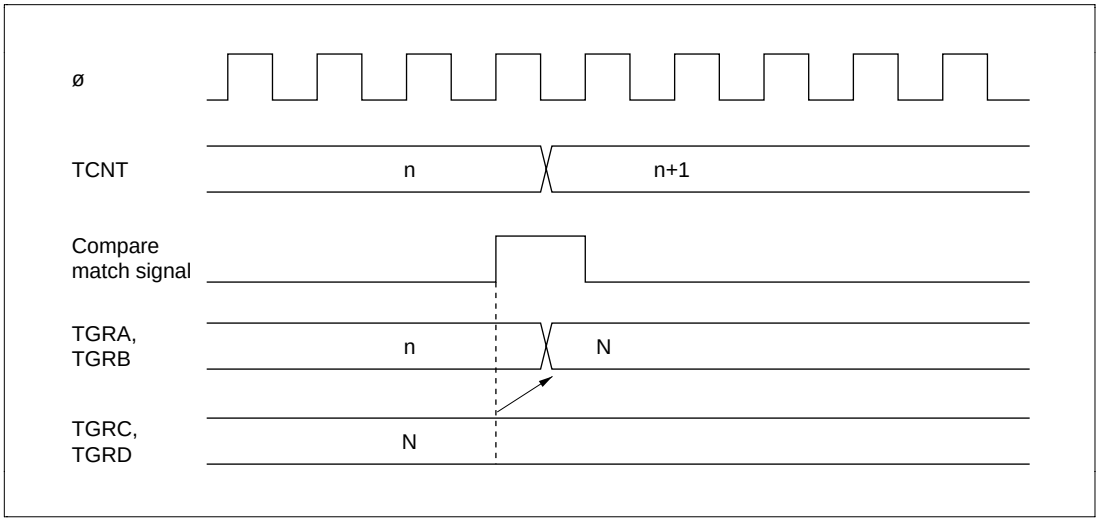



**Figure 10-40 Buffer Operation Timing (Compare Match)**

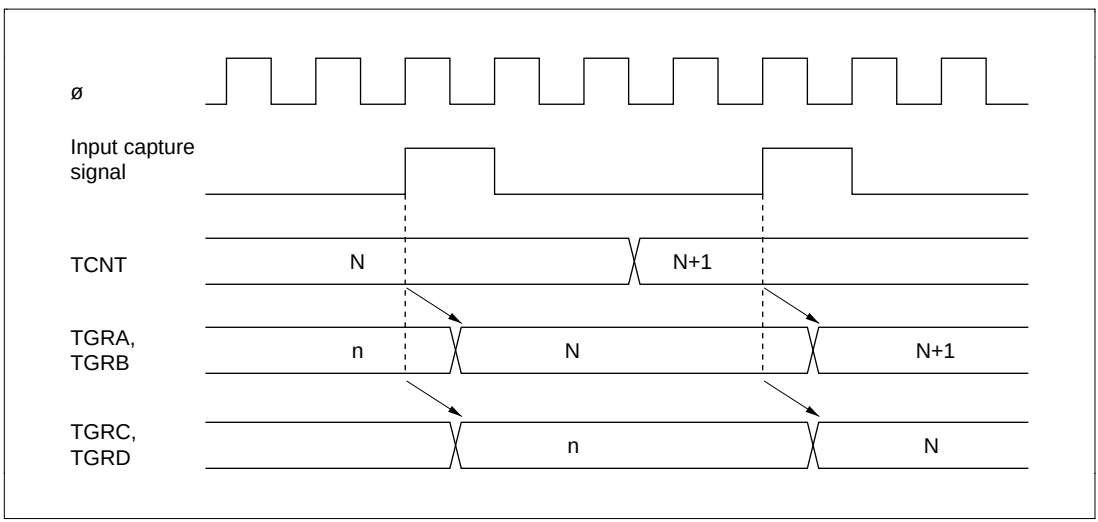



**Figure 10-41 Buffer Operation Timing (Input Capture)**

### 10.6.2 Interrupt Signal Timing

**TGF Flag Setting Timing in Case of Compare Match:** Figure 10-42 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and TGI interrupt request signal timing.

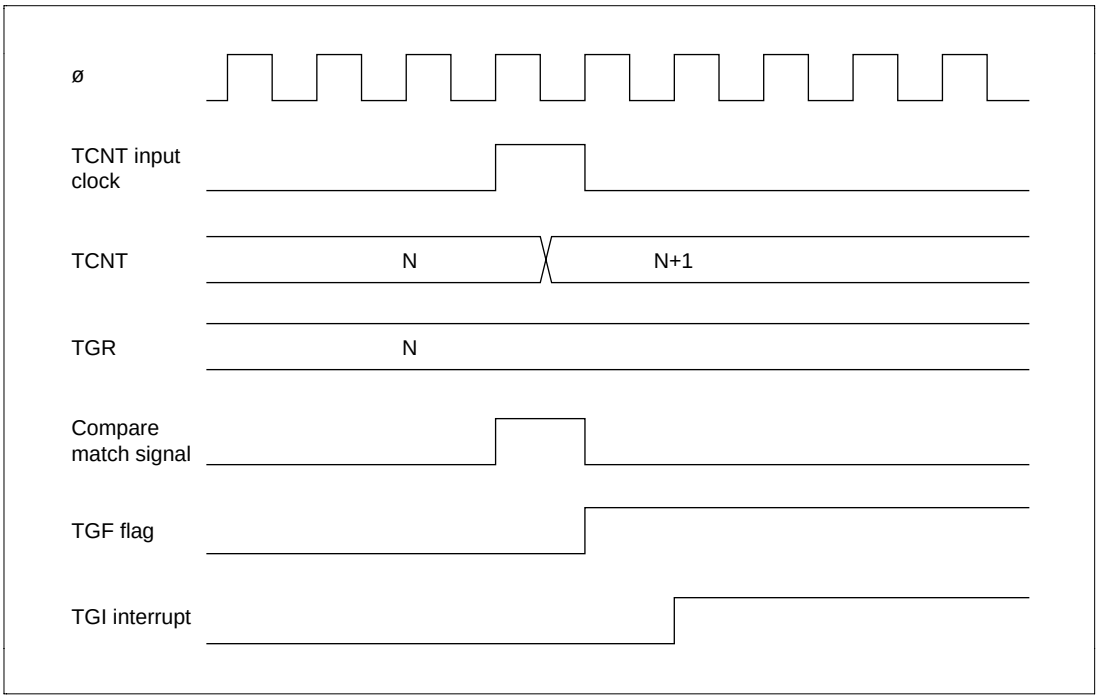



**Figure 10-42 TGI Interrupt Timing (Compare Match)**

**TGF Flag Setting Timing in Case of Input Capture:** Figure 10-43 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and TGI interrupt request signal timing.

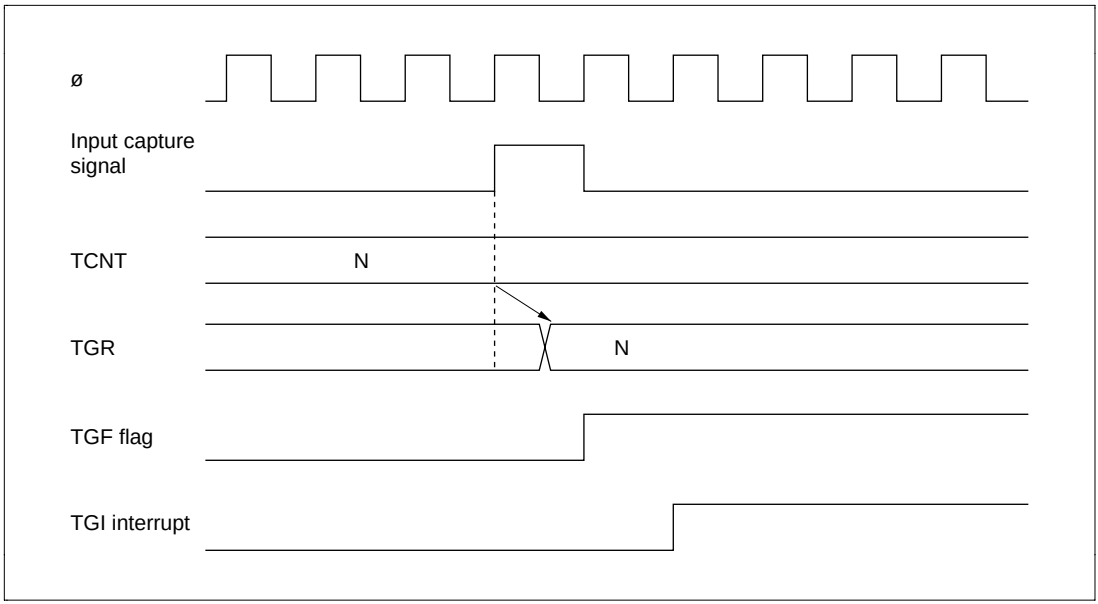



**Figure 10-43 TGI Interrupt Timing (Input Capture)**

**TCFV Flag/TCFU Flag Setting Timing:** Figure 10-44 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and TCIV interrupt request signal timing.

Figure 10-45 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and TCIU interrupt request signal timing.

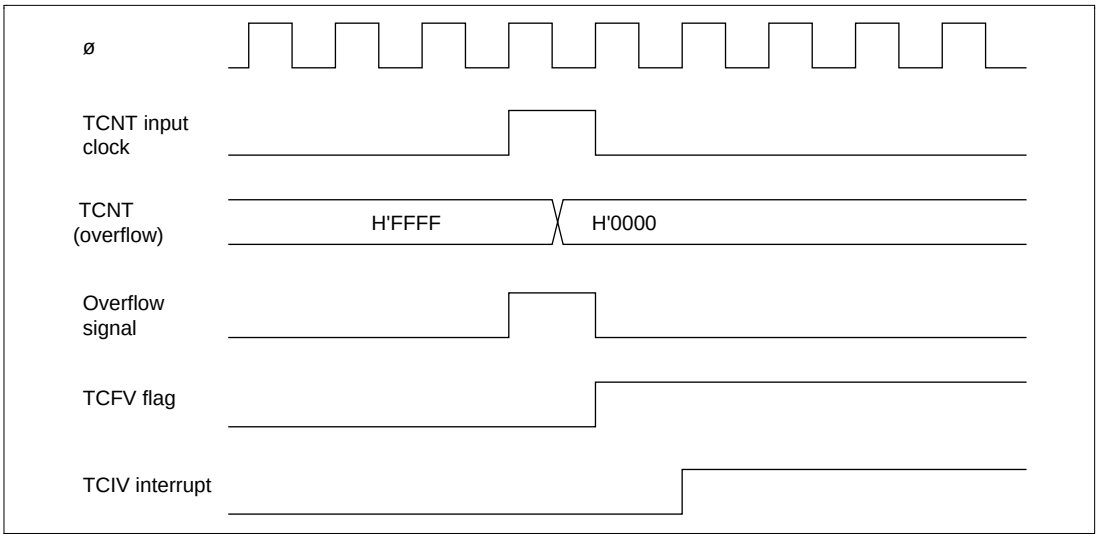



**Figure 10-44 TCIV Interrupt Setting Timing**

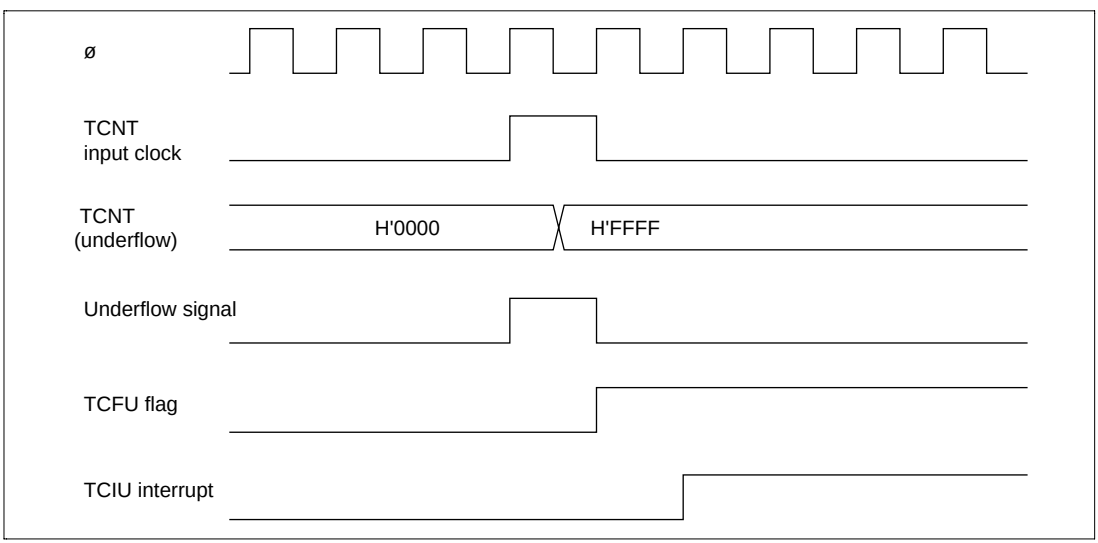



**Figure 10-45 TCIU Interrupt Setting Timing**

**Status Flag Clearing Timing:** After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DTC or DMAC is activated, the flag is cleared automatically. Figure 10-46 shows the timing for status flag clearing by the CPU, and figure 10-47 shows the timing for status flag clearing by the DTC or DMAC.

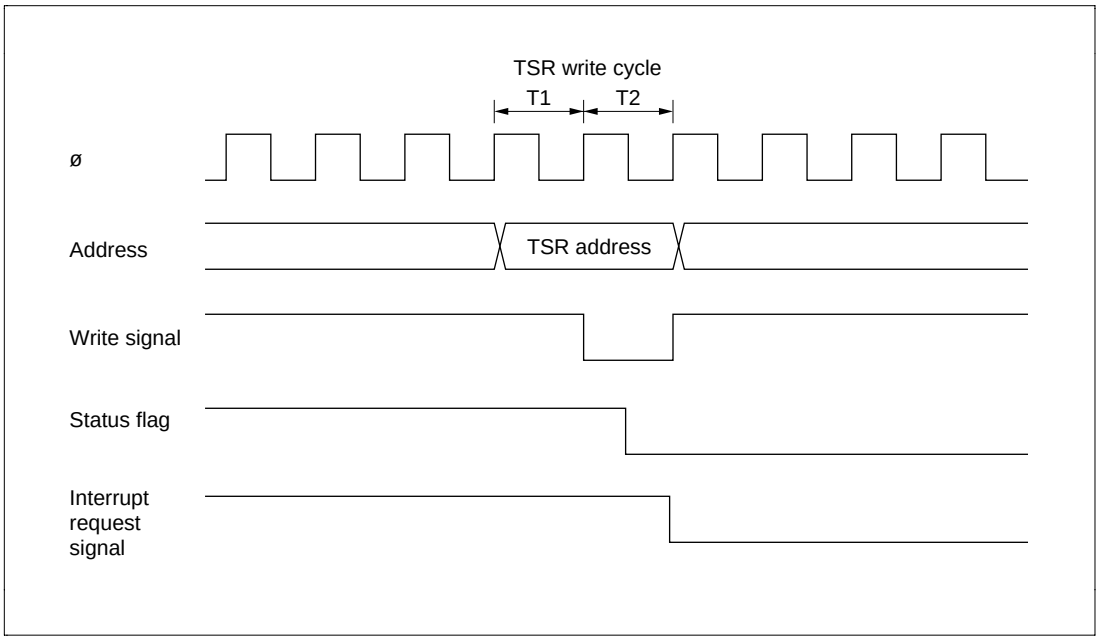



**Figure 10-46 Timing for Status Flag Clearing by CPU**

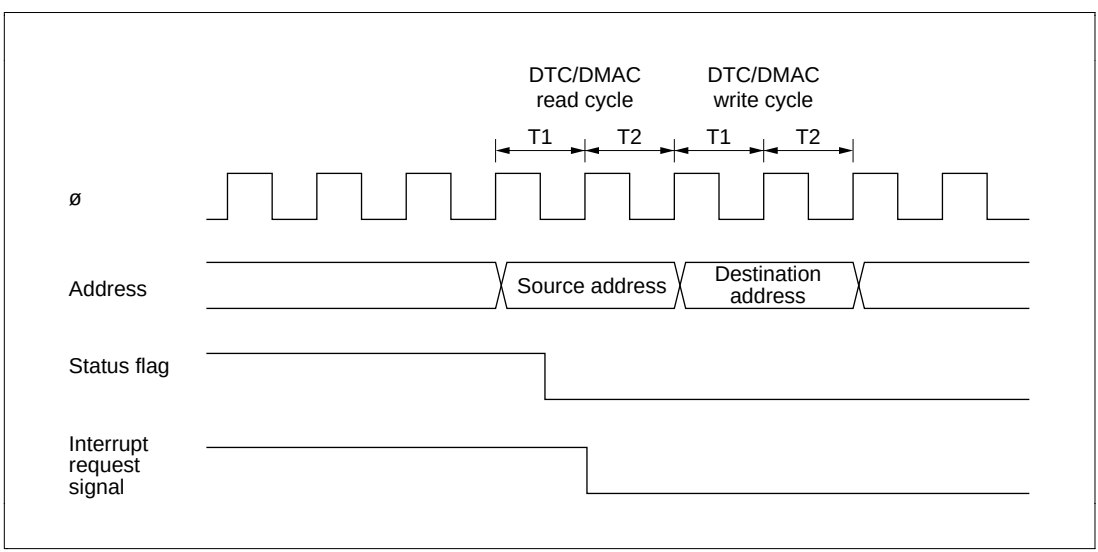



**Figure 10-47 Timing for Status Flag Clearing by DTC/DMAC Activation**

## 10.7 Usage Notes

Note that the kinds of operation and contention described below occur during TPU operation.

**Input Clock Restrictions:** The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 10-48 shows the input clock conditions in phase counting mode.

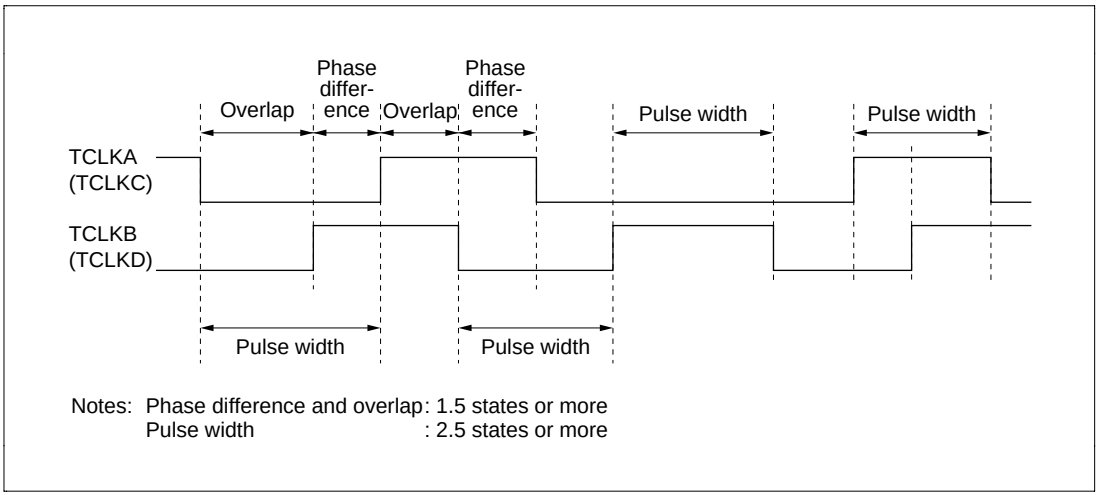



**Figure 10-48 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode**

**Caution on Period Setting:** When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

$$f = \frac{\phi}{(N + 1)}$$

Where f : Counter frequency
ø : Operating frequency
N : TGR set value

**Contention between TCNT Write and Clear Operations:** If the counter clear signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed.

Figure 10-49 shows the timing in this case.

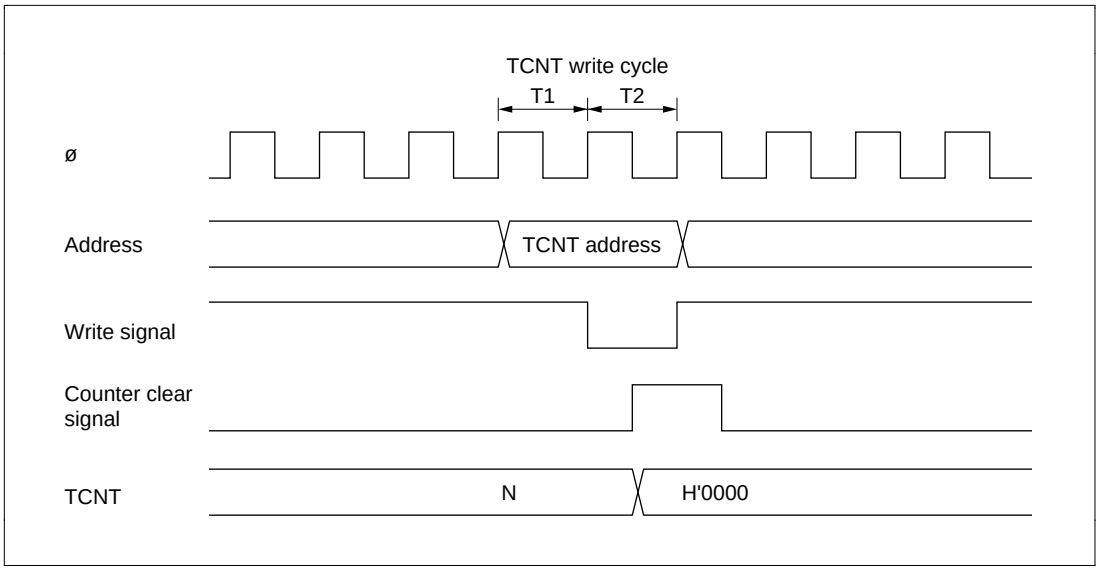



**Figure 10-49 Contention between TCNT Write and Clear Operations**

**Contention between TCNT Write and Increment Operations:** If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented.

Figure 10-50 shows the timing in this case.

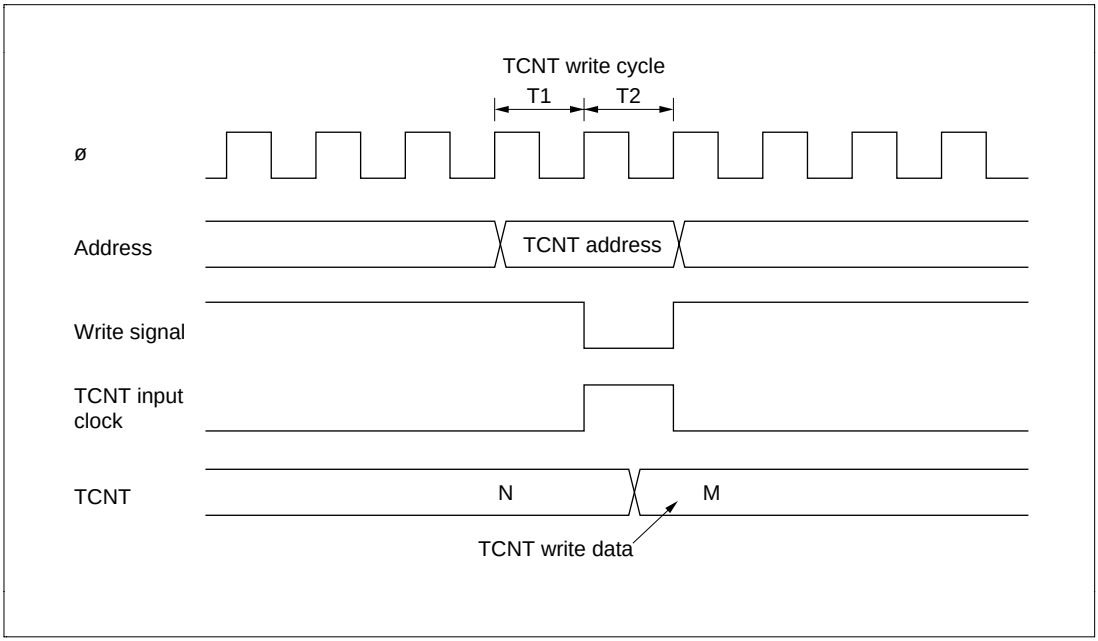



**Figure 10-50 Contention between TCNT Write and Increment Operations**

**Contention between TGR Write and Compare Match:** If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is inhibited. A compare match does not occur even if the same value as before is written.

Figure 10-51 shows the timing in this case.

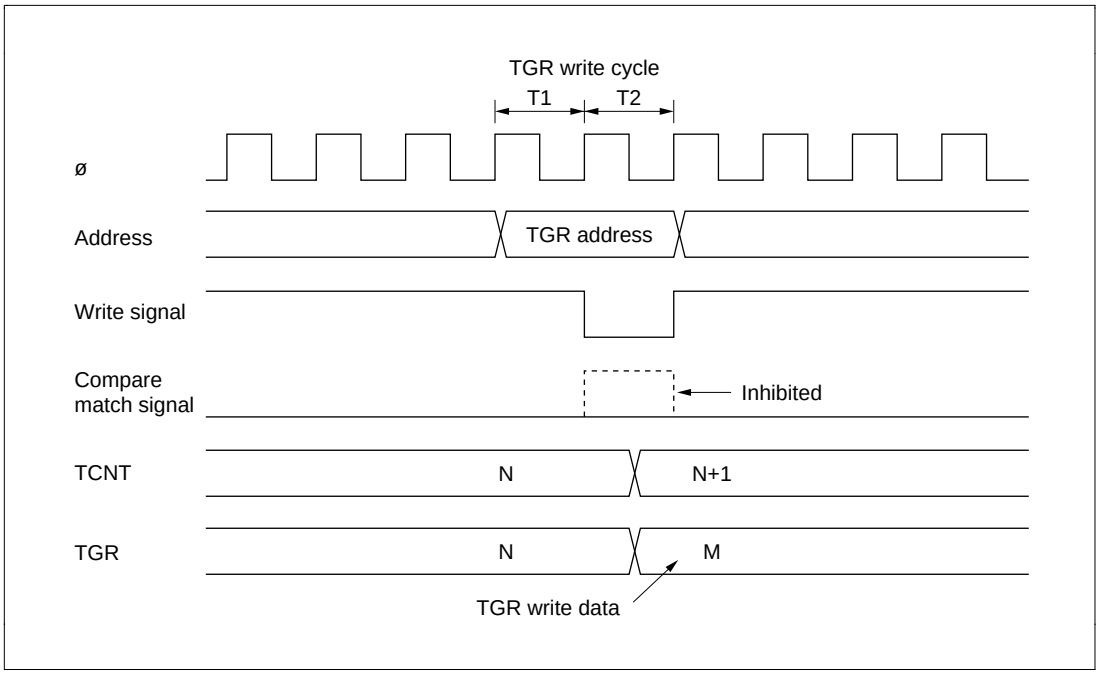



**Figure 10-51 Contention between TGR Write and Compare Match**

**Contention between Buffer Register Write and Compare Match:** If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the data prior to the write.

Figure 10-52 shows the timing in this case.

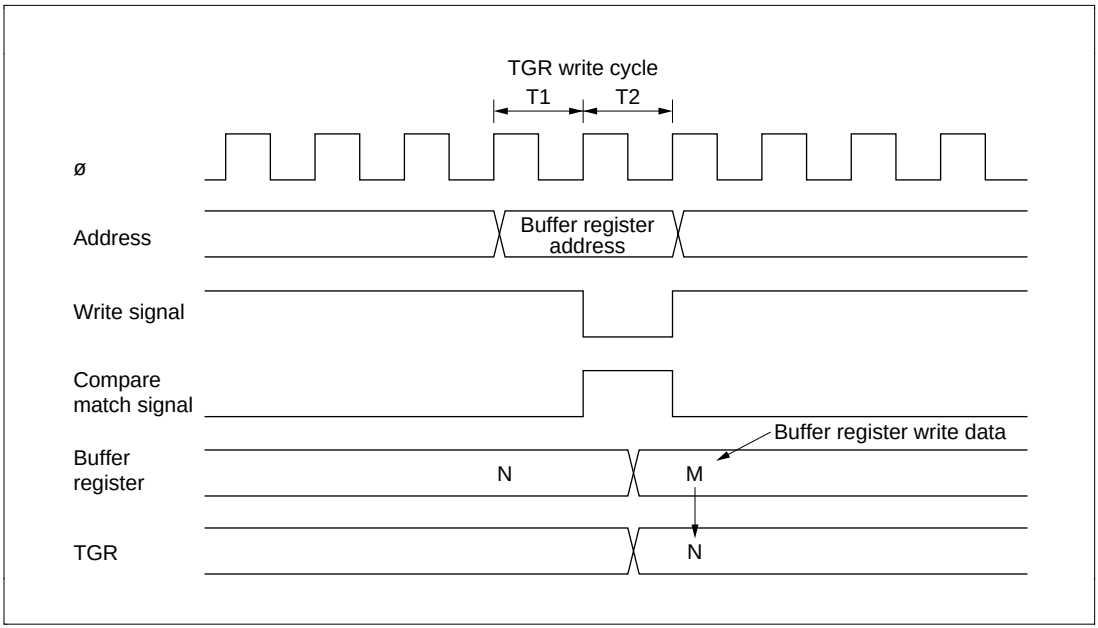



**Figure 10-52 Contention between Buffer Register Write and Compare Match**

**Contention between TGR Read and Input Capture:** If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 10-53 shows the timing in this case.

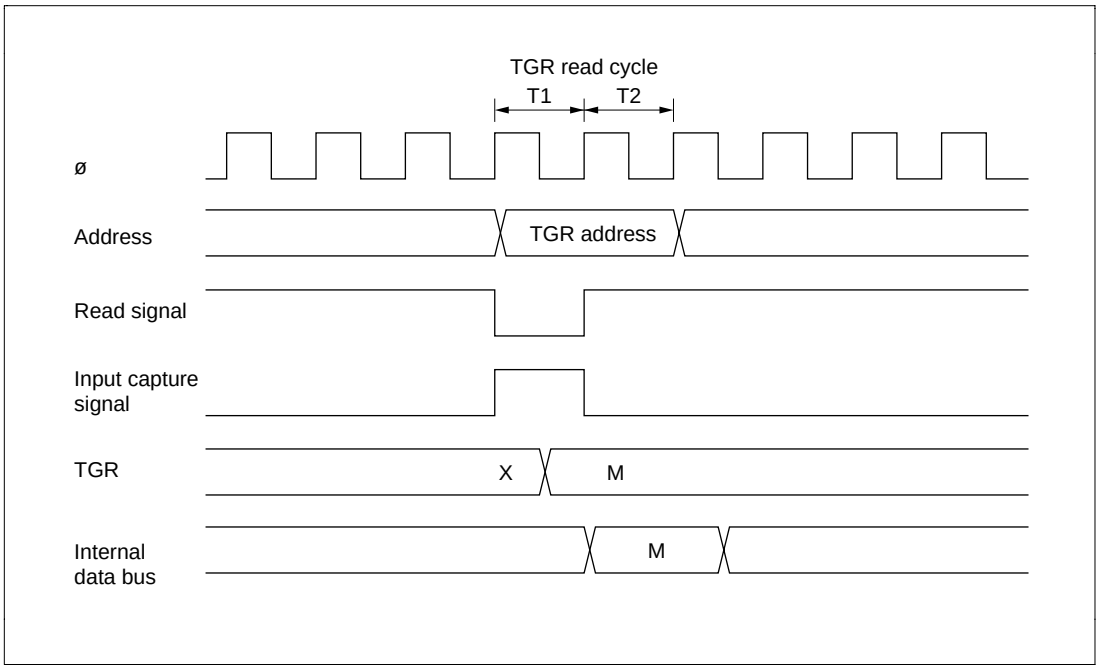



**Figure 10-53 Contention between TGR Read and Input Capture**

**Contention between TGR Write and Input Capture:** If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 10-54 shows the timing in this case.

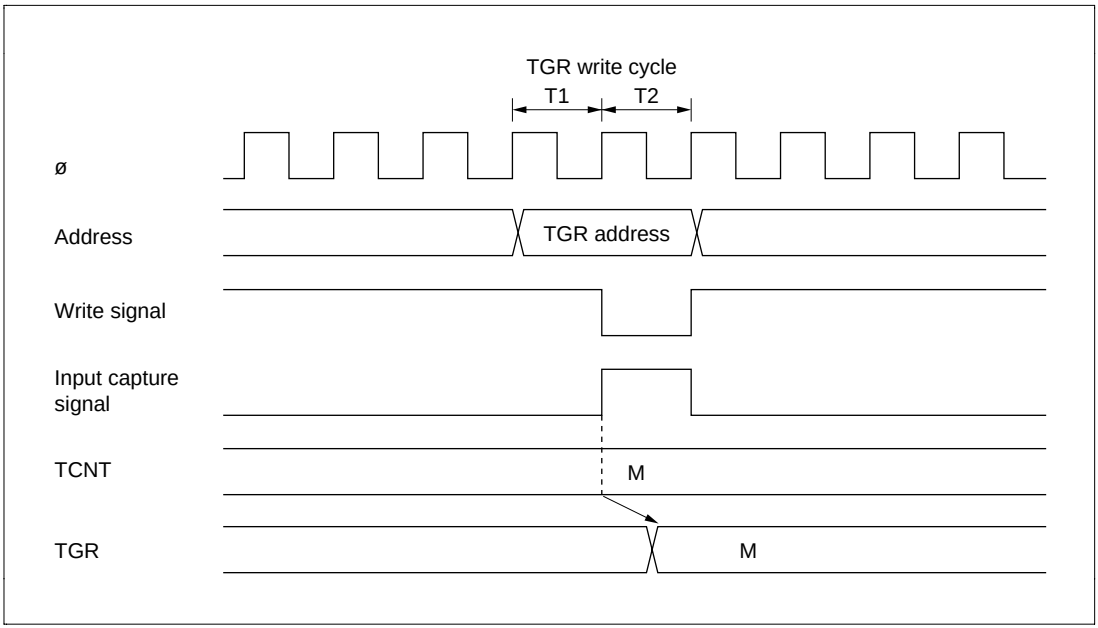



**Figure 10-54 Contention between TGR Write and Input Capture**

**Contention between Buffer Register Write and Input Capture:** If the input capture signal is generated in the T2 state of a buffer write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 10-55 shows the timing in this case.

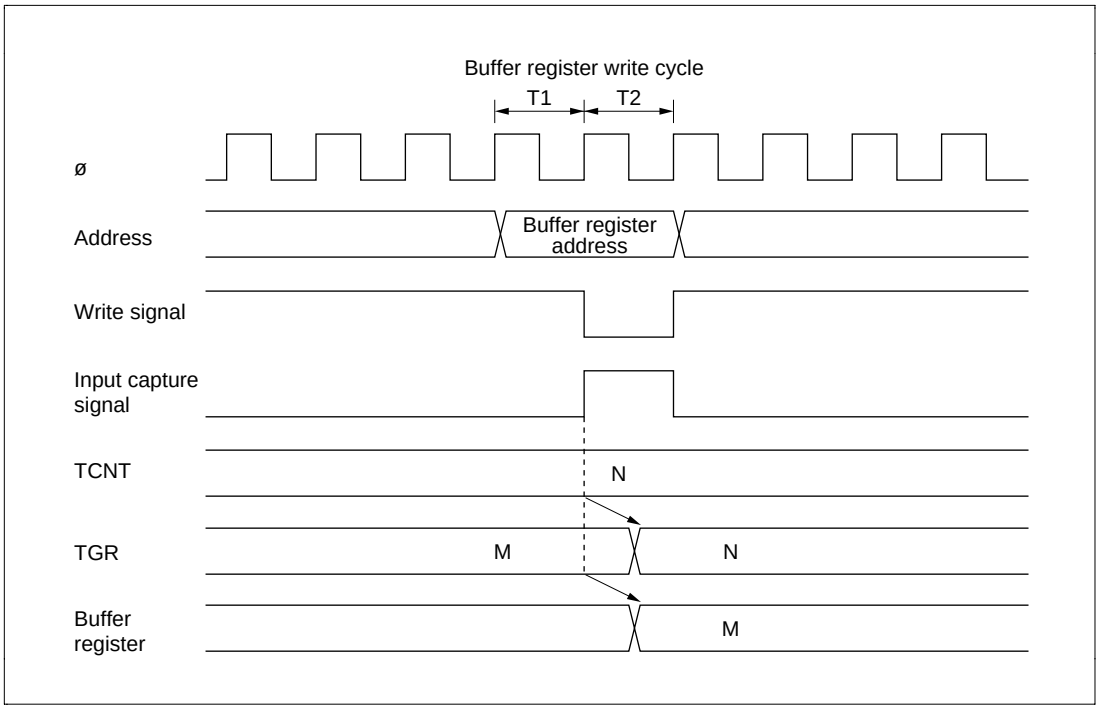



**Figure 10-55 Contention between Buffer Register Write and Input Capture**

**Contention between Overflow/Underflow and Counter Clearing:** If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 10-56 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.

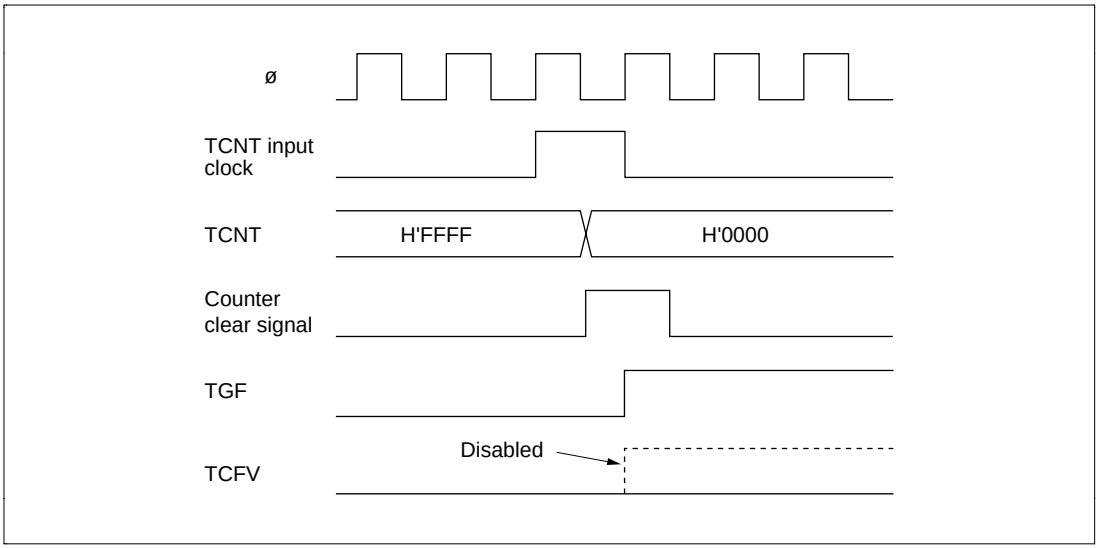



**Figure 10-56 Contention between Overflow and Counter Clearing**

**Contention between TCNT Write and Overflow/Underflow:** If there is an up-count or down-count in the T2 state of a TCNT write cycle, and overflow/underflow occurs, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 10-57 shows the operation timing when there is contention between TCNT write and overflow.

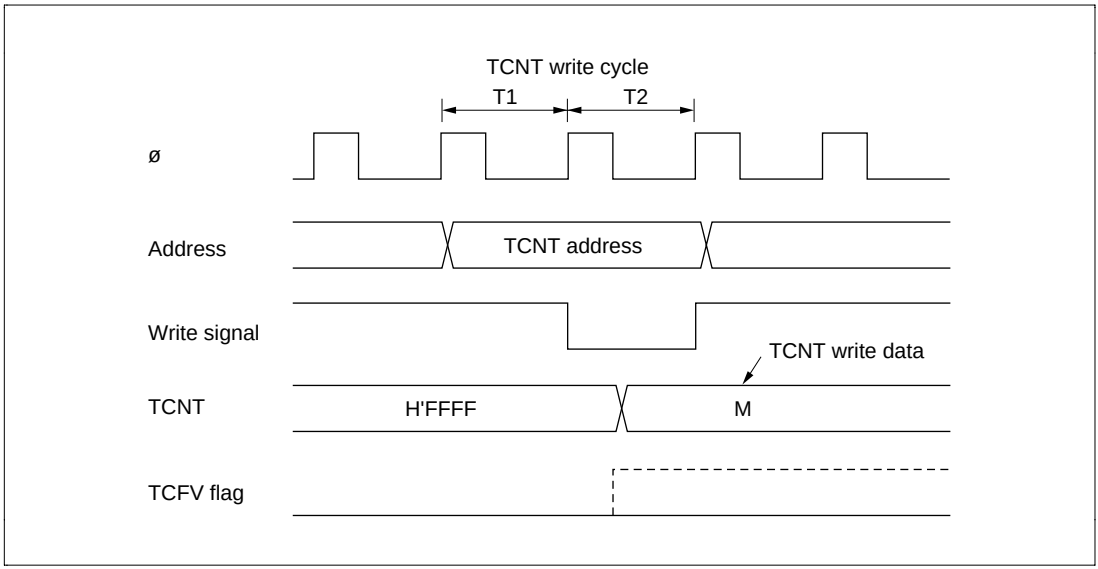



**Figure 10-57 Contention between TCNT Write and Overflow**

**Multiplexing of I/O Pins:** In the H8S/2655 Series, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

**Interrupts and Module Stop Mode:** If module stop mode is set when an interrupt has been requested, the CPU interrupt source or DMAC/DTC activation source cannot be cleared. Interrupts should therefore be disabled before setting module stop mode.

# Section 11 Programmable Pulse Generator (PPG)

## 11.1 Overview

The H8S/2655 Series has a built-in programmable pulse generator (PPG) that provides pulse outputs by using the 16-bit timer-pulse unit (TPU) as a time base. The PPG pulse outputs are divided into 4-bit groups (group 3 to group 0) that can operate both simultaneously and independently.

### 11.1.1 Features

PPG features are listed below.

- 16-bit output data
  Maximum 16-bit data can be output, and output can be enabled on a bit-by-bit basis.
- Four output groups
  Output trigger signals can be selected in 4-bit groups to provide up to four different 4-bit outputs.
- Selectable output trigger signals
  Output trigger signals can be selected for each group from the compare match signals of four TPU channels.
- Non-overlap mode
  A non-overlap margin can be provided between pulse outputs.
- Can operate together with the data transfer controller (DTC) and DMA controller (DMAC).
  The compare match signals selected as output trigger signals can activate the DTC or DMAC for sequential output of data without CPU intervention.
- Settable inverted output
  Inverted data can be output for each group.

### 11.1.2 Block Diagram

Figure 11-1 shows a block diagram of the PPG.

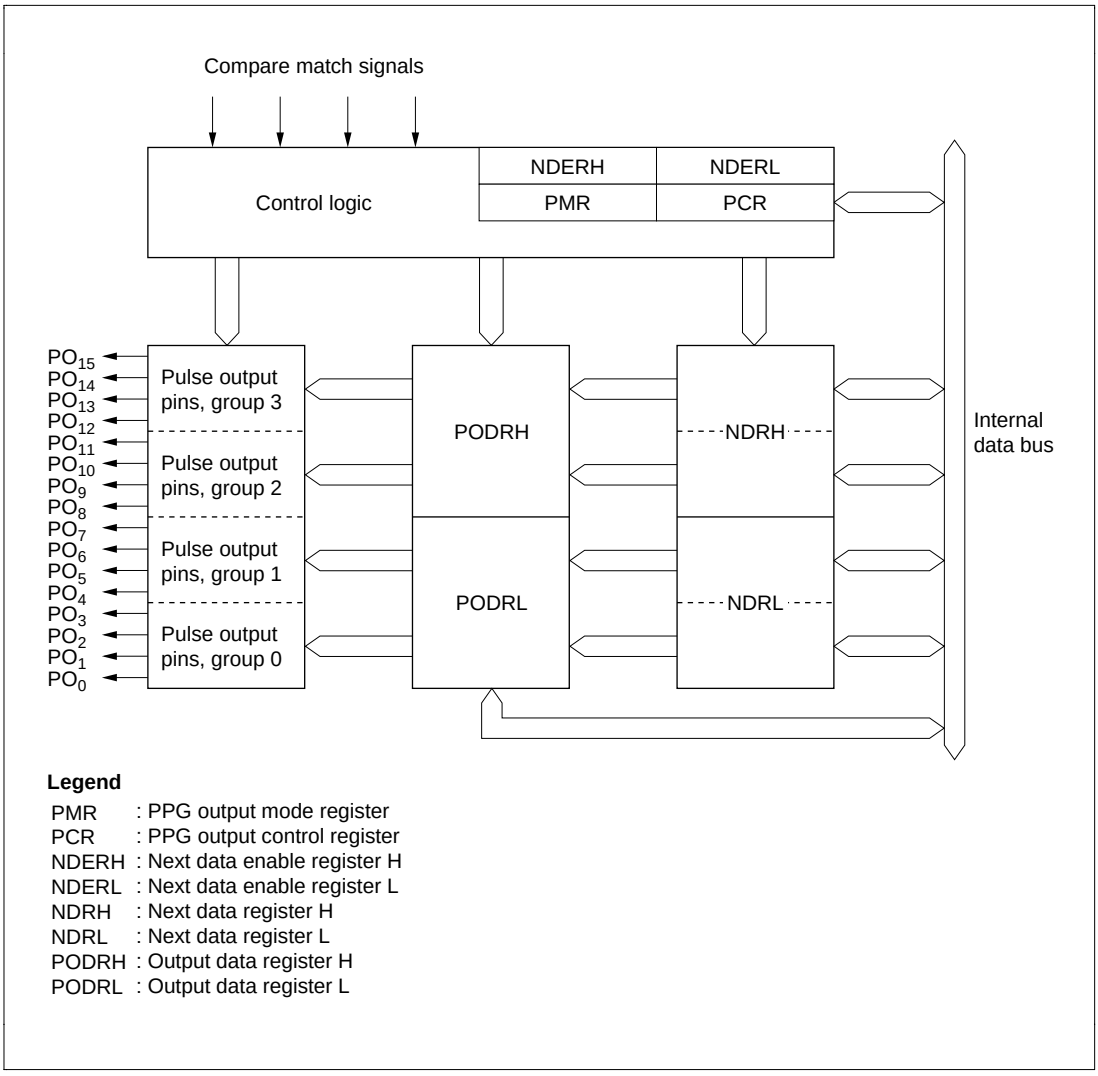



**Figure 11-1 Block Diagram of PPG**

### 11.1.3 Pin Configuration

Table 11-1 summarizes the PPG pins.

**Table 11-1 PPG Pins**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Pulse output 0 | $PO_0$ | Output | Group 0 pulse output |
| Pulse output 1 | $PO_1$ | Output | |
| Pulse output 2 | $PO_2$ | Output | |
| Pulse output 3 | $PO_3$ | Output | |
| Pulse output 4 | $PO_4$ | Output | Group 1 pulse output |
| Pulse output 5 | $PO_5$ | Output | |
| Pulse output 6 | $PO_6$ | Output | |
| Pulse output 7 | $PO_7$ | Output | |
| Pulse output 8 | $PO_8$ | Output | Group 2 pulse output |
| Pulse output 9 | $PO_9$ | Output | |
| Pulse output 10 | $PO_{10}$ | Output | |
| Pulse output 11 | $PO_{11}$ | Output | |
| Pulse output 12 | $PO_{12}$ | Output | Group 3 pulse output |
| Pulse output 13 | $PO_{13}$ | Output | |
| Pulse output 14 | $PO_{14}$ | Output | |
| Pulse output 15 | $PO_{15}$ | Output | |

### 11.1.4 Registers

Table 11-2 summarizes the PPG registers.

**Table 11-2 PPG Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| PPG output control register | PCR | R/W | H'FF | H'FF46 |
| PPG output mode register | PMR | R/W | H'F0 | H'FF47 |
| Next data enable register H | NDERH | R/W | H'00 | H'FF48 |
| Next data enable register L | NDERL | R/W | H'00 | H'FF49 |
| Output data register H | PODRH | R/(W)*[2] | H'00 | H'FF4A |
| Output data register L | PODRL | R/(W)*[2] | H'00 | H'FF4B |
| Next data register H | NDRH | R/W | H'00 | H'FF4C*[3]<br>H'FF4E |
| Next data register L | NDRL | R/W | H'00 | H'FF4D*[3]<br>H'FF4F |
| Port 1 data direction register | P1DDR | W | H'00 | H'FEB0 |
| Port 2 data direction register | P2DDR | W | H'00 | H'FEB1 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.

2. Bits used for pulse output cannot be written to.

3. When the same output trigger is selected for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FF4C. When the output triggers are different, the NDRH address is H'FF4E for group 2 and H'FF4C for group 3.

   Similarly, when the same output trigger is selected for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FF4D. When the output triggers are different, the NDRL address is H'FF4F for group 0 and H'FF4D for group 1.

## 11.2 Register Descriptions

### 11.2.1 Next Data Enable Registers H and L (NDERH, NDERL)

**NDERH**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDER15 | NDER14 | NDER13 | NDER12 | NDER11 | NDER10 | NDER9 | NDER8 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**NDERL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDER7 | NDER6 | NDER5 | NDER4 | NDER3 | NDER2 | NDER1 | NDER0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

NDERH and NDERL are 8-bit readable/writable registers that enable or disable pulse output on a bit-by-bit basis.

If a bit is enabled for pulse output by NDERH or NDERL, the NDR value is automatically transferred to the corresponding PODR bit when the TPU compare match event specified by PCR occurs, updating the output value. If pulse output is disabled, the bit value is not transferred from NDR to PODR and the output value does not change.

NDERH and NDERL are each initialized to H'00 by a reset and in hardware standby mode. They are not initialized in software standby mode.

**NDERH Bits 7 to 0—Next Data Enable 15 to 8 (NDER15 to NDER8):** These bits enable or disable pulse output on a bit-by-bit basis.

**Bits 7 to 0**

| NDER15 to NDER8 | Description |
|---|---|
| 0 | Pulse outputs $PO_{15}$ to $PO_8$ are disabled (NDR15 to NDR8 are not transferred to POD15 to POD8) (Initial value) |
| 1 | Pulse outputs $PO_{15}$ to $PO_8$ are enabled (NDR15 to NDR8 are transferred to POD15 to POD8) |

**NDERL Bits 7 to 0—Next Data Enable 7 to 0 (NDER7 to NDER0):** These bits enable or disable pulse output on a bit-by-bit basis.

**Bits 7 to 0**

| NDER7 to NDER0 | Description |
|---|---|
| 0 | Pulse outputs $PO_7$ to $PO_0$ are disabled (NDR7 to NDR0 are not transferred to POD7 to POD0) (Initial value) |
| 1 | Pulse outputs $PO_7$ to $PO_0$ are enabled (NDR7 to NDR0 are transferred to POD7 to POD0) |

### 11.2.2 Output Data Registers H and L (PODRH, PODRL)

**PODRH**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | POD15 | POD14 | POD13 | POD12 | POD11 | POD10 | POD9 | POD8 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

**PODRL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | POD7 | POD6 | POD5 | POD4 | POD3 | POD2 | POD1 | POD0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * A bit that has been set for pulse output by NDER is read-only.

PODRH and PODRL are 8-bit readable/writable registers that store output data for use in pulse output.

### 11.2.3 Next Data Registers H and L (NDRH, NDRL)

NDRH and NDRL are 8-bit readable/writable registers that store the next data for pulse output. During pulse output, the contents of NDRH and NDRL are transferred to the corresponding bits in PODRH and PODRL when the TPU compare match event specified by PCR occurs. The NDRH and NDRL addresses differ depending on whether pulse output groups have the same output trigger or different output triggers. For details see section 11.2.4, Notes on NDR Access.

NDRH and NDRL are each initialized to H'00 by a reset and in hardware standby mode. They are not initialized in software standby mode.

### 11.2.4 Notes on NDR Access

The NDRH and NDRL addresses differ depending on whether pulse output groups have the same output trigger or different output triggers.

**Same Trigger for Pulse Output Groups:** If pulse output groups 2 and 3 are triggered by the same compare match event, the NDRH address is H'FF4C. The upper 4 bits belong to group 3 and the lower 4 bits to group 2. Address H'FF4E consists entirely of reserved bits that cannot be modified and are always read as 1.

**Address H'FF4C**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDR15 | NDR14 | NDR13 | NDR12 | NDR11 | NDR10 | NDR9 | NDR8 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Address H'FF4E**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | — | — | — |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | — | — | — | — | — | — | — | — |

If pulse output groups 0 and 1 are triggered by the same compare match event, the NDRL address is H'FF4D. The upper 4 bits belong to group 1 and the lower 4 bits to group 0. Address H'FF4F consists entirely of reserved bits that cannot be modified and are always read as 1.

**Address H'FF4D**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Address H'FF4F**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | — | — | — |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | — | — | — | — | — | — | — | — |

**Different Triggers for Pulse Output Groups:** If pulse output groups 2 and 3 are triggered by different compare match events, the address of the upper 4 bits in NDRH (group 3) is H'FF4C and the address of the lower 4 bits (group 2) is H'FF4E. Bits 3 to 0 of address H'FF4C and bits 7 to 4 of address H'FF4E are reserved bits that cannot be modified and are always read as 1.

**Address H'FF4C**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDR15 | NDR14 | NDR13 | NDR12 | — | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | — | — | — | — |

**Address H'FF4E**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | NDR11 | NDR10 | NDR9 | NDR8 |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

If pulse output groups 0 and 1 are triggered by different compare match event, the address of the upper 4 bits in NDRL (group 1) is H'FF4D and the address of the lower 4 bits (group 0) is H'FF4F. Bits 3 to 0 of address H'FF4D and bits 7 to 4 of address H'FF4F are reserved bits that cannot be modified and are always read as 1.

**Address H'FF4D**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDR7 | NDR6 | NDR5 | NDR4 | — | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | — | — | — | — |

**Address H'FF4F**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

### 11.2.5 PPG Output Control Register (PCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | G3CMS1 | G3CMS0 | G2CMS1 | G2CMS0 | G1CMS1 | G1CMS0 | G0CMS1 | G0CMS0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PCR is an 8-bit readable/writable register that selects output trigger signals for PPG outputs on a group-by-group basis.

PCR is initialized to H'FF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 7 and 6—Group 3 Compare Match Select 1 and 0 (G3CMS1, G3CMS0):** These bits select the compare match that triggers pulse output group 3 (pins $PO_{15}$ to $PO_{12}$).

| Bit 7 | Bit 6 | Description | |
|---|---|---|---|
| **G3CMS1** | **G3CMS0** | **Output Trigger for Pulse Output Group 3** | |
| 0 | 0 | Compare match in TPU channel 0 | |
| | 1 | Compare match in TPU channel 1 | |
| 1 | 0 | Compare match in TPU channel 2 | |
| | 1 | Compare match in TPU channel 3 | (Initial value) |

**Bits 5 and 4—Group 2 Compare Match Select 1 and 0 (G2CMS1, G2CMS0):** These bits select the compare match that triggers pulse output group 2 (pins $PO_{11}$ to $PO_{8}$).

| Bit 5 | Bit 4 | Description | |
|---|---|---|---|
| **G2CMS1** | **G2CMS0** | **Output Trigger for Pulse Output Group 2** | |
| 0 | 0 | Compare match in TPU channel 0 | |
| | 1 | Compare match in TPU channel 1 | |
| 1 | 0 | Compare match in TPU channel 2 | |
| | 1 | Compare match in TPU channel 3 | (Initial value) |

**Bits 3 and 2—Group 1 Compare Match Select 1 and 0 (G1CMS1, G1CMS0):** These bits select the compare match that triggers pulse output group 1 (pins $PO_7$ to $PO_4$).

| Bit 3 | Bit 2 | Description | |
|---|---|---|---|
| **G1CMS1** | **G1CMS0** | **Output Trigger for Pulse Output Group 1** | |
| 0 | 0 | Compare match in TPU channel 0 | |
| | 1 | Compare match in TPU channel 1 | |
| 1 | 0 | Compare match in TPU channel 2 | |
| | 1 | Compare match in TPU channel 3 | (Initial value) |

**Bits 1 and 0—Group 0 Compare Match Select 1 and 0 (G0CMS1, G0CMS0):** These bits select the compare match that triggers pulse output group 0 (pins $PO_3$ to $PO_0$).

| Bit 1 | Bit 0 | Description | |
|---|---|---|---|
| **G0CMS1** | **G0CMS0** | **Output Trigger for Pulse Output Group 0** | |
| 0 | 0 | Compare match in TPU channel 0 | |
| | 1 | Compare match in TPU channel 1 | |
| 1 | 0 | Compare match in TPU channel 2 | |
| | 1 | Compare match in TPU channel 3 | (Initial value) |

### 11.2.6 PPG Output Mode Register (PMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | G3INV | G2INV | G1INV | G0INV | G3NOV | G2NOV | G1NOV | G0NOV |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PMR is an 8-bit readable/writable register that selects pulse output inversion and non-overlapping operation for each group.

The output trigger period of a non-overlapping operation PPG output waveform is set in TGRB and the non-overlap margin is set in TGRA. The output values change at compare match A and B.

For details, see section 11.3.4, Non-Overlapping Pulse Output.

PMR is initialized to H'F0 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Group 3 Inversion (G3INV):** Selects direct output or inverted output for pulse output group 3 (pins $PO_{15}$ to $PO_{12}$).

**Bit 7**

| G3INV | Description |
|---|---|
| 0 | Inverted output for pulse output group 3 (low-level output at pin for a 1 in PODRH) |
| 1 | Direct output for pulse output group 3 (high-level output at pin for a 1 in PODRH) (Initial value) |

**Bit 6—Group 2 Inversion (G2INV):** Selects direct output or inverted output for pulse output group 2 (pins $PO_{11}$ to $PO_{8}$).

**Bit 6**

| G2INV | Description |
|---|---|
| 0 | Inverted output for pulse output group 2 (low-level output at pin for a 1 in PODRH) |
| 1 | Direct output for pulse output group 2 (high-level output at pin for a 1 in PODRH) (Initial value) |

**Bit 5—Group 1 Inversion (G1INV):** Selects direct output or inverted output for pulse output group 1 (pins $PO_7$ to $PO_4$).

**Bit 5**

| G1INV | Description |
|---|---|
| 0 | Inverted output for pulse output group 1 (low-level output at pin for a 1 in PODRL) |
| 1 | Direct output for pulse output group 1 (high-level output at pin for a 1 in PODRL) (Initial value) |

**Bit 4—Group 0 Inversion (G0INV):** Selects direct output or inverted output for pulse output group 0 (pins $PO_3$ to $PO_0$).

**Bit 4**

| G0INV | Description |
|---|---|
| 0 | Inverted output for pulse output group 0 (low-level output at pin for a 1 in PODRL) |
| 1 | Direct output for pulse output group 0 (high-level output at pin for a 1 in PODRL) (Initial value) |

**Bit 3—Group 3 Non-Overlap (G3NOV):** Selects normal or non-overlapping operation for pulse output group 3 (pins $PO_{15}$ to $PO_{12}$).

**Bit 3**

| G3NOV | Description |
|---|---|
| 0 | Normal operation in pulse output group 3 (output values updated at compare match A in the selected TPU channel) (Initial value) |
| 1 | Non-overlapping operation in pulse output group 3 (independent 1 and 0 output at compare match A or B in the selected TPU channel) |

**Bit 2—Group 2 Non-Overlap (G2NOV):** Selects normal or non-overlapping operation for pulse output group 2 (pins $PO_{11}$ to $PO_8$).

**Bit 2**

| G2NOV | Description |
|---|---|
| 0 | Normal operation in pulse output group 2 (output values updated at compare match A in the selected TPU channel) (Initial value) |
| 1 | Non-overlapping operation in pulse output group 2 (independent 1 and 0 output at compare match A or B in the selected TPU channel) |

**Bit 1—Group 1 Non-Overlap (G1NOV):** Selects normal or non-overlapping operation for pulse output group 1 (pins $PO_7$ to $PO_4$).

**Bit 1**

| G1NOV | Description |
|---|---|
| 0 | Normal operation in pulse output group 1 (output values updated at compare match A in the selected TPU channel) (Initial value) |
| 1 | Non-overlapping operation in pulse output group 1 (independent 1 and 0 output at compare match A or B in the selected TPU channel) |

**Bit 0—Group 0 Non-Overlap (G0NOV):** Selects normal or non-overlapping operation for pulse output group 0 (pins $PO_3$ to $PO_0$).

**Bit 0**

| G0NOV | Description |
|---|---|
| 0 | Normal operation in pulse output group 0 (output values updated at compare match A in the selected TPU channel) (Initial value) |
| 1 | Non-overlapping operation in pulse output group 0 (independent 1 and 0 output at compare match A or B in the selected TPU channel) |

### 11.2.7 Port 1 Data Direction Register (P1DDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P1DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 1.

Port 1 is multiplexed with pins $PO_{15}$ to $PO_8$. Bits corresponding to pins used for PPG output must be set to 1. For further information about P1DDR, see section 9.2, Port 1.

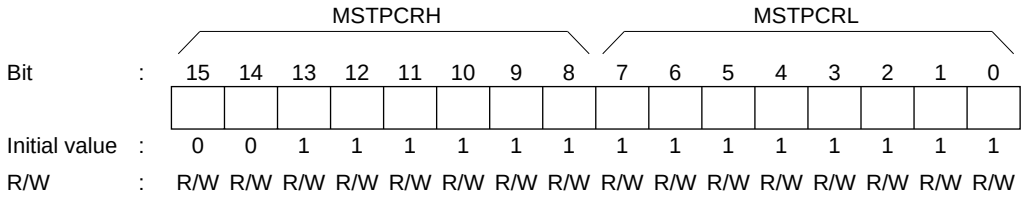
### 11.2.8 Port 2 Data Direction Register (P2DDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P2DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 2.

Port 2 is multiplexed with pins $PO_7$ to $PO_0$. Bits corresponding to pins used for PPG output must be set to 1. For further information about P2DDR, see section 9.3, Port 2.

### 11.2.9 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP11 bit in MSTPCR is set to 1, PPG operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 11—Module Stop (MSTP11):** Specifies the PPG module stop mode.

**Bit 11**

| MSTP11 | Description | |
|---|---|---|
| 0 | PPG module stop mode cleared | |
| 1 | PPG module stop mode set | (Initial value) |

## 11.3 Operation

### 11.3.1 Overview

PPG pulse output is enabled when the corresponding bits in P1DDR, P2DDR, and NDER are set to 1. In this state the corresponding PODR contents are output.

When the compare match event specified by PCR occurs, the corresponding NDR bit contents are transferred to PODR to update the output values.

Figure 11-2 illustrates the PPG output operation and table 11-3 summarizes the PPG operating conditions.

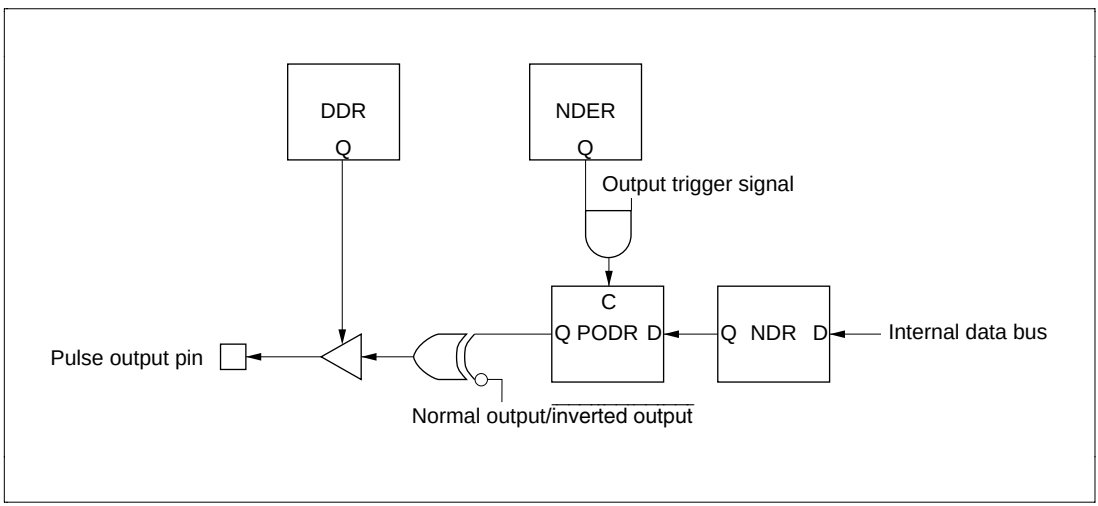



**Figure 11-2 PPG Output Operation**

**Table 11-3 PPG Operating Conditions**

| NDER | DDR | Pin Function |
|---|---|---|
| 0 | 0 | Generic input port |
| | 1 | Generic output port |
| 1 | 0 | Generic input port (but the PODR bit is a read-only bit, and when compare match occurs, the NDR bit value is transferred to the PODR bit) |
| | 1 | PPG pulse output |

Sequential output of data of up to 16 bits is possible by writing new output data to NDR before the next compare match. For details of non-overlapping operation, see section 11.3.4, Non-Overlapping Pulse Output.

### 11.3.2 Output Timing

If pulse output is enabled, NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 11-3 shows the timing of these operations for the case of normal output in groups 2 and 3, triggered by compare match A.

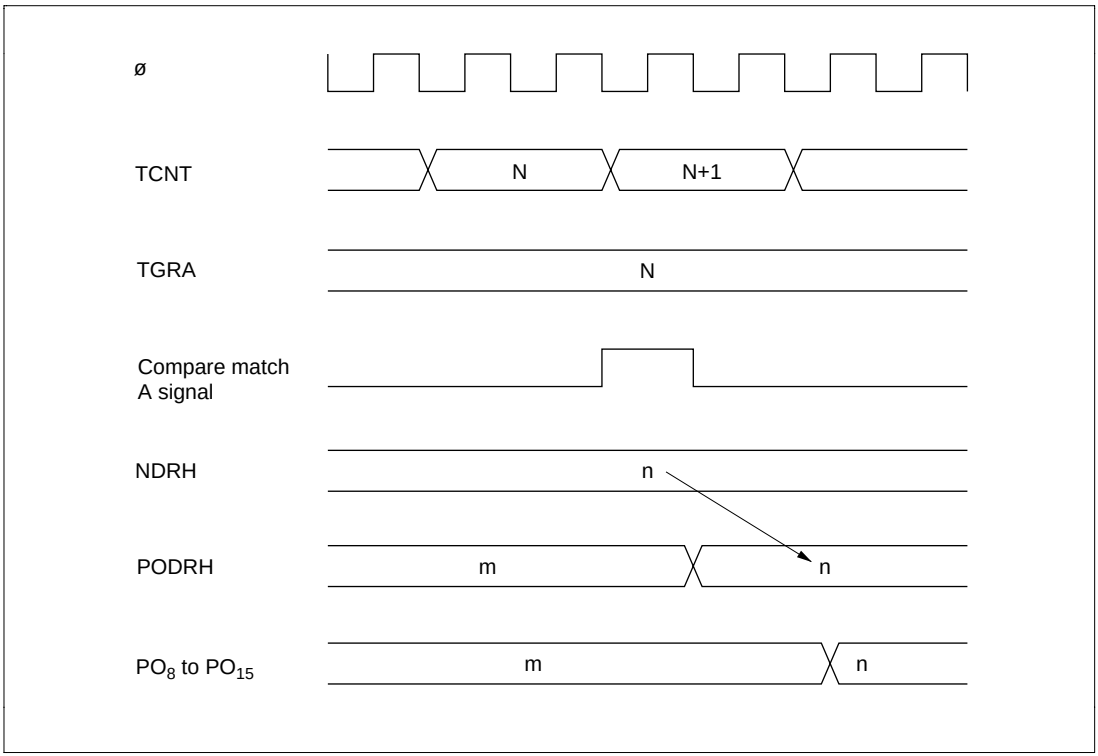
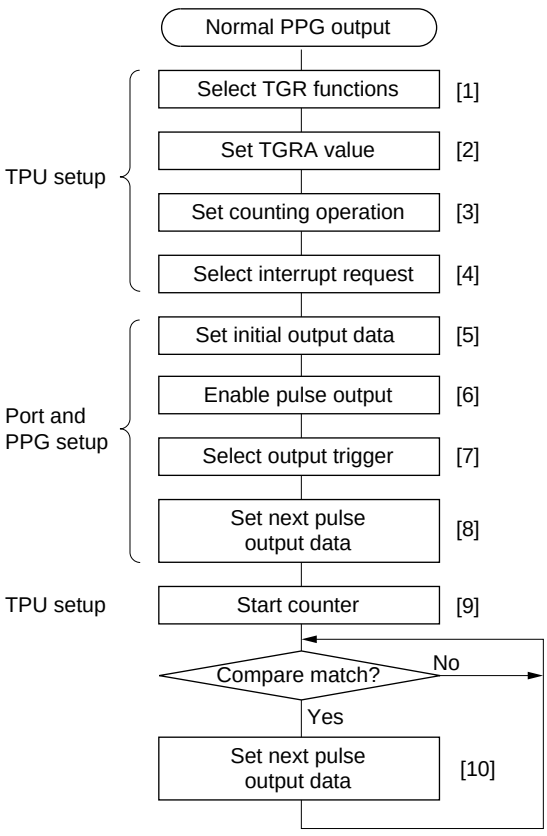



**Figure 11-3 Timing of Transfer and Output of NDR Contents (Example)**

### 11.3.3 Normal Pulse Output

**Sample Setup Procedure for Normal Pulse Output:** Figure 11-4 shows a sample procedure for setting up normal pulse output.




[1] Set TIOR to make TGRA an output compare register (with output disabled)

[2] Set the PPG output trigger period

[3] Select the counter clock source with bits TPSC2 to TPSC0 in TCR. Select the counter clear source with bits CCLR1 and CCLR0.

[4] Enable the TGIA interrupt in TIER. The DTC or DMAC can also be set up to transfer data to NDR.

[5] Set the initial output values in PODR.

[6] Set the DDR and NDER bits for the pins to be used for pulse output to 1.

[7] Select the TPU compare match event to be used as the output trigger in PCR.

[8] Set the next pulse output values in NDR.

[9] Set the CST bit in TSTR to 1 to start the TCNT counter.

[10] At each TGIA interrupt, set the next output values in NDR.

**Figure 11-4 Setup Procedure for Normal Pulse Output (Example)**

**Example of Normal Pulse Output (Example of Five-Phase Pulse Output):** Figure 11-5 shows an example in which pulse output is used for cyclic five-phase pulse output.

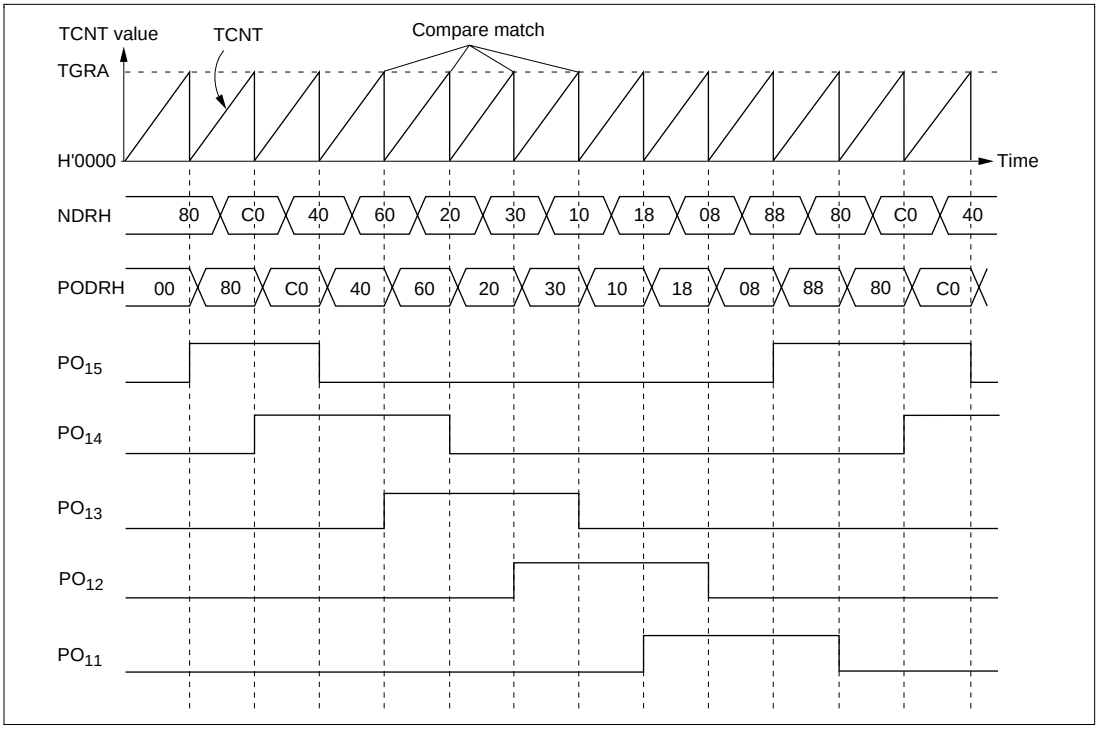



**Figure 11-5 Normal Pulse Output Example (Five-Phase Pulse Output)**

[1] Set up the TPU channel to be used as the output trigger channel so that TGRA is an output compare register and the counter will be cleared by compare match A. Set the trigger period in TGRA and set the TGIEA bit in TIER to 1 to enable the compare match A (TGIA) interrupt.

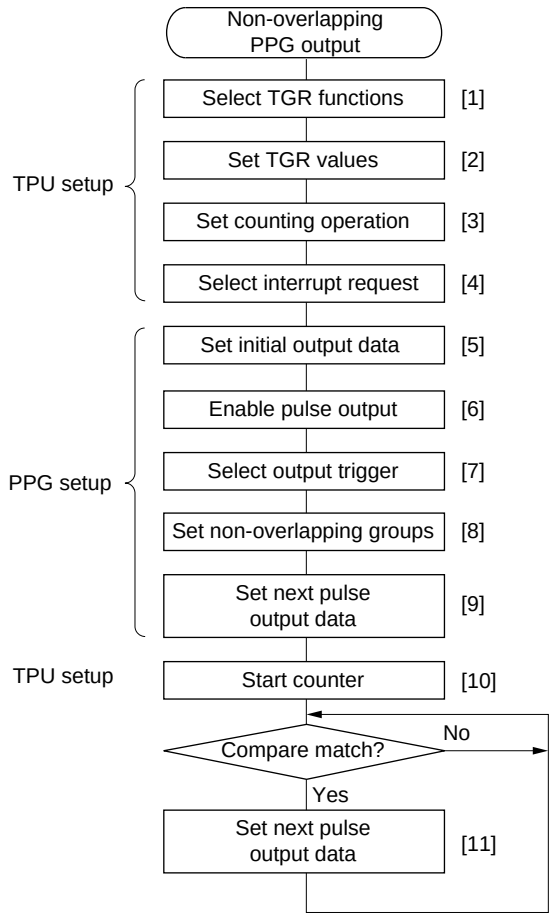
[2] Write H'F8 in P1DDR and NDERH, and set the G3CMS1, G3CMS0, G2CMS1, and G2CMS0 bits in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Write output data H'80 in NDRH.

[3] The timer counter in the TPU channel starts. When compare match A occurs, the NDRH contents are transferred to PODRH and output. The TGIA interrupt handling routine writes the next output data (H'C0) in NDRH.

[4] Five-phase overlapping pulse output (one or two phases active at a time) can be obtained subsequently by writing H'40, H'60, H'20, H'30. H'10, H'18, H'08, H'88... at successive TGIA interrupts. If the DTC or DMAC is set for activation by this interrupt, pulse output can be obtained without imposing a load on the CPU.

### 11.3.4 Non-Overlapping Pulse Output

**Sample Setup Procedure for Non-Overlapping Pulse Output:** Figure 11-6 shows a sample procedure for setting up non-overlapping pulse output.




[1] Set TIOR to make TGRA and TGRB an output compare registers (with output disabled)

[2] Set the pulse output trigger period in TGRB and the non-overlap margin in TGRA.

[3] Select the counter clock source with bits TPSC2 to TPSC0 in TCR. Select the counter clear source with bits CCLR1 and CCLR0.

[4] Enable the TGIA interrupt in TIER. The DTC or DMAC can also be set up to transfer data to NDR.

[5] Set the initial output values in PODR.

[6] Set the DDR and NDER bits for the pins to be used for pulse output to 1.

[7] Select the TPU compare match event to be used as the pulse output trigger in PCR.

[8] In PMR, select the groups that will operate in non-overlap mode.

[9] Set the next pulse output values in NDR.

[10] Set the CST bit in TSTR to 1 to start the TCNT counter.

[11] At each TGIA interrupt, set the next output values in NDR.

**Figure 11-6 Setup Procedure for Non-Overlapping Pulse Output (Example)**

**Example of Non-Overlapping Pulse Output (Example of Four-Phase Complementary Non-Overlapping Output):** Figure 11-7 shows an example in which pulse output is used for four-phase complementary non-overlapping pulse output.

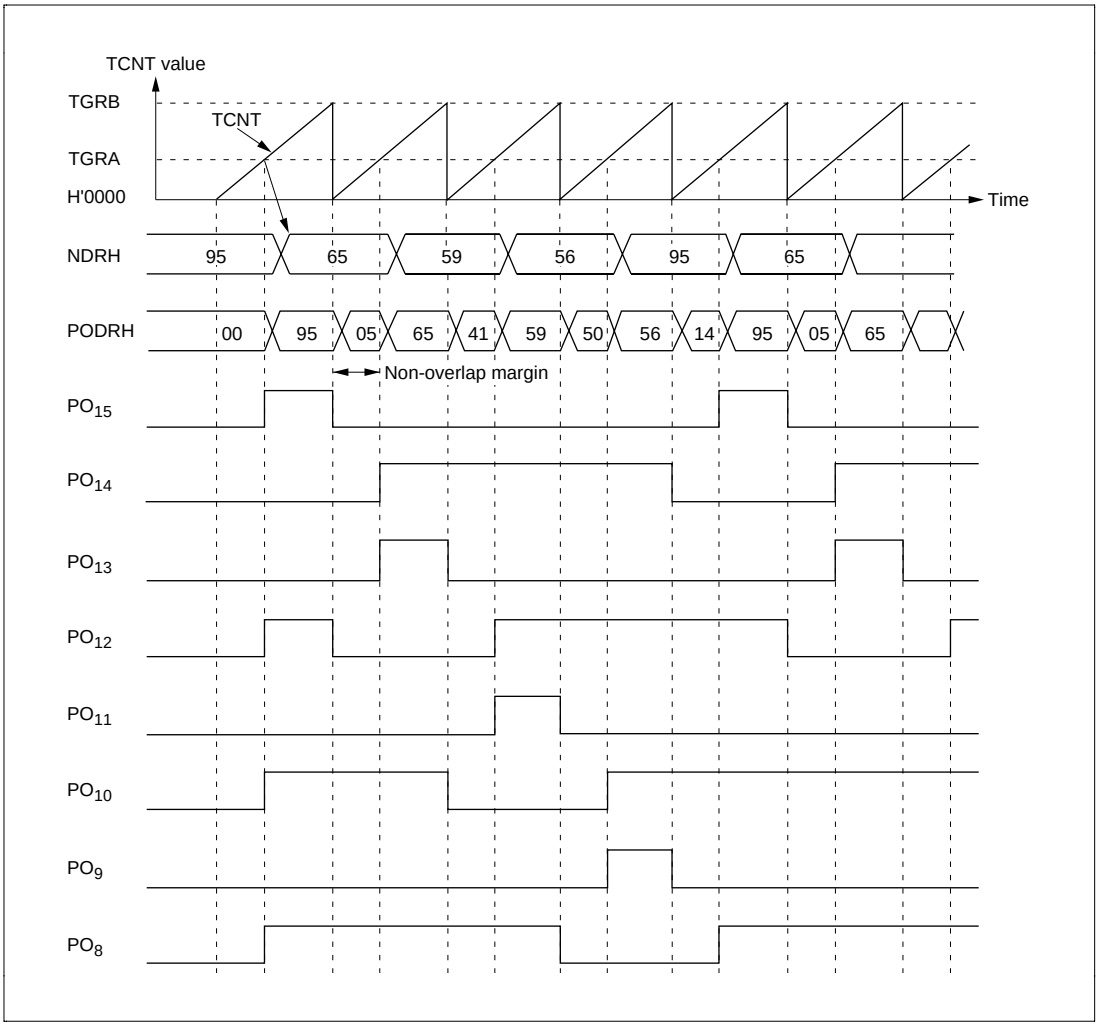



**Figure 11-7 Non-Overlapping Pulse Output Example (Four-Phase Complementary)**

[1] Set up the TPU channel to be used as the output trigger channel so that TGRA and TGRB are output compare registers. Set the trigger period in TGRB and the non-overlap margin in TGRA, and set the counter to be cleared by compare match B. Set the TGIEA bit in TIER to 1 to enable the TGIA interrupt.

[2] Write H'FF in P1DDR and NDERH, and set the G3CMS1, G3CMS0, G2CMS1, and G2CMS0 bits in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Set the G3NOV and G2NOV bits in PMR to 1 to select non-overlapping output. Write output data H'95 in NDRH.

[3] The timer counter in the TPU channel starts. When a compare match with TGRB occurs, outputs change from 1 to 0. When a compare match with TGRA occurs, outputs change from 0 to 1 (the change from 0 to 1 is delayed by the value set in TGRA). The TGIA interrupt handling routine writes the next output data (H'65) in NDRH.

[4] Four-phase complementary non-overlapping pulse output can be obtained subsequently by writing H'59, H'56, H'95... at successive TGIA interrupts. If the DTC or DMAC is set for activation by this interrupt, pulse output can be obtained without imposing a load on the CPU.

### 11.3.5 Inverted Pulse Output

If the G3INV, G2INV, G1INV, and G0INV bits in PMR are cleared to 0, values that are the inverse of the PODR contents can be output.

Figure 11-8 shows the outputs when G3INV and G2INV are cleared to 0, in addition to the settings of figure 11-7.

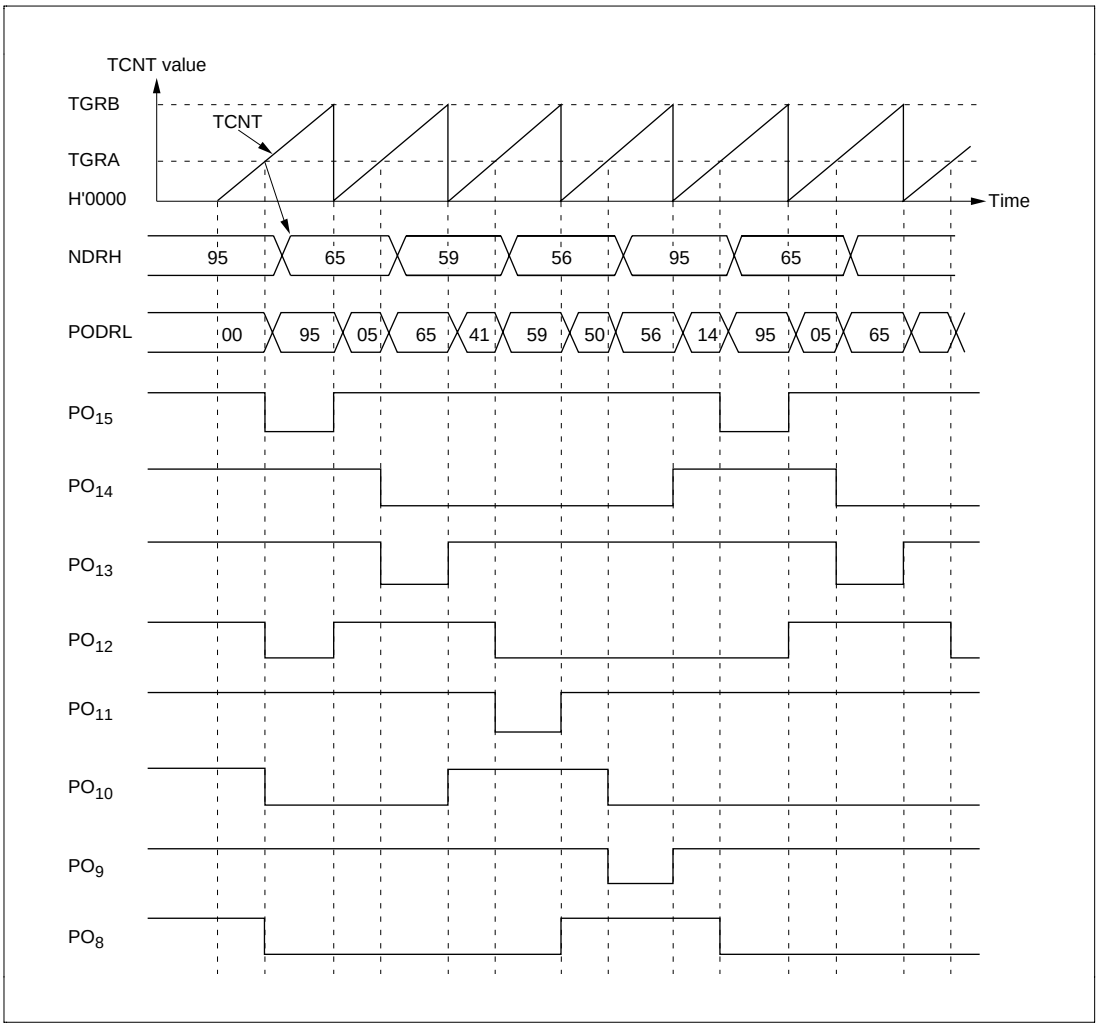



**Figure 11-8 Inverted Pulse Output (Example)**

### 11.3.6 Pulse Output Triggered by Input Capture

Pulse output can be triggered by TPU input capture as well as by compare match. If TGRA functions as an input capture register in the TPU channel selected by PCR, pulse output will be triggered by the input capture signal.

Figure 11-9 shows the timing of this output.

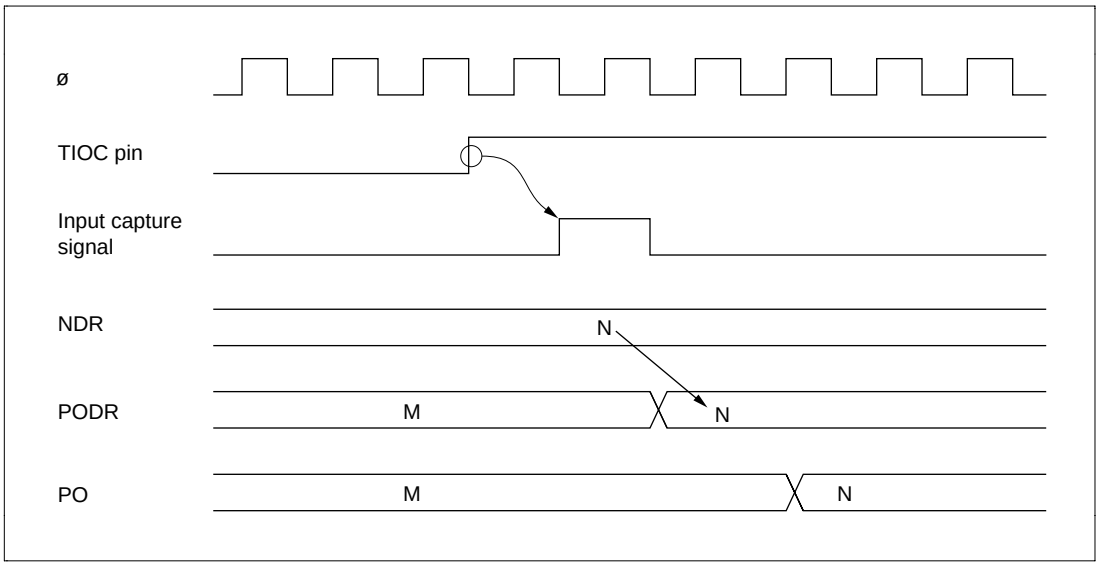



**Figure 11-9 Pulse Output Triggered by Input Capture (Example)**

## 11.4 Usage Notes

### 11.4.1 Operation of Pulse Output Pins

Pins $PO_0$ to $PO_{15}$ are also used for other peripheral functions such as the TPU. When output by another peripheral function is enabled, the corresponding pins cannot be used for pulse output. Note, however, that data transfer from NDR bits to PODR bits takes place, regardless of the usage of the pins.

Pin functions should be changed only under conditions in which the output trigger event will not occur.

### 11.4.2 Note on Non-Overlapping Output

During non-overlapping operation, the transfer of NDR bit values to PODR bits takes place as follows.

- NDR bits are always transferred to PODR bits at compare match A.
- At compare match B, NDR bits are transferred only if their value is 0. Bits are not transferred if their value is 1.

Figure 11-10 illustrates the non-overlapping pulse output operation.

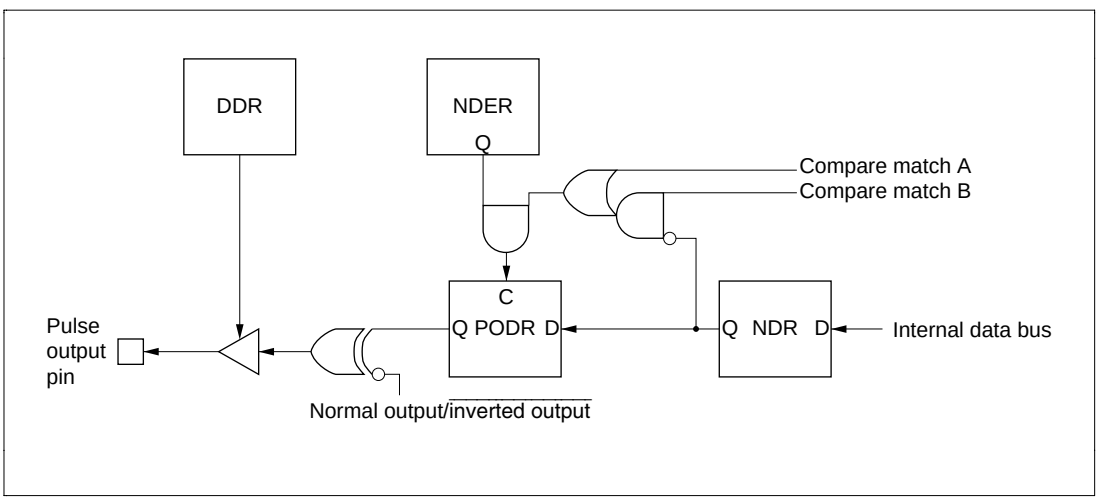



**Figure 11-10 Non-Overlapping Pulse Output**

Therefore, 0 data can be transferred ahead of 1 data by making compare match B occur before compare match A. The NDR contents should not be altered during the interval from compare match B to compare match A (the non-overlap margin).

This can be accomplished by having the TGIA interrupt handling routine write the next data in NDR, or by having the TGIA interrupt activate the DTC or DMAC. Note, however, that the next data must be written before the next compare match B occurs.

Figure 11-11 shows the timing of this operation.

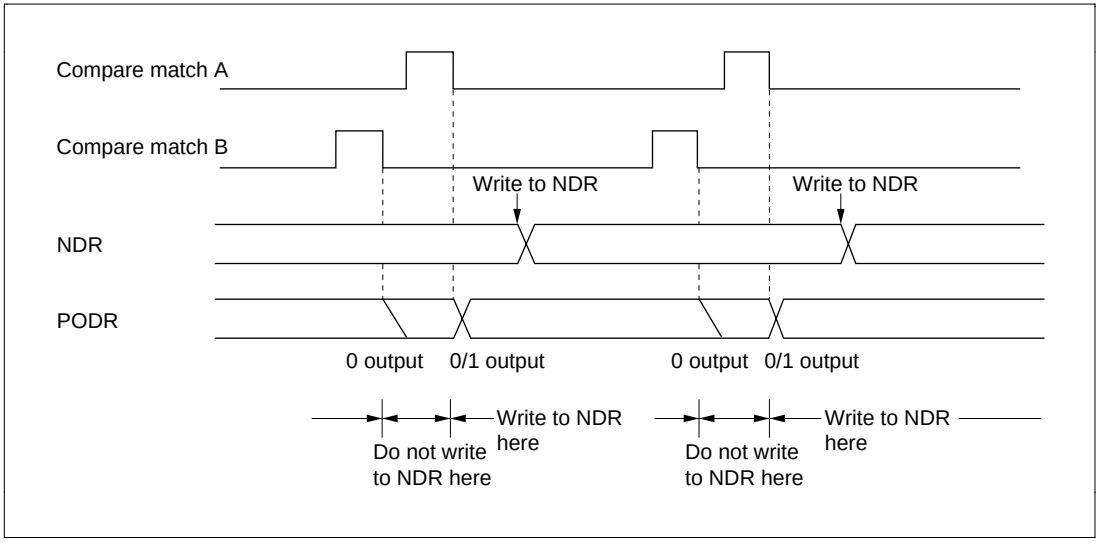



**Figure 11-11 Non-Overlapping Operation and NDR Write Timing**

# Section 12 8-Bit Timers

## 12.1 Overview

The H8S/2655 Series includes an 8-bit timer module with two channels (TMR0 and TMR1). Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare match events. The 8-bit timer module can thus be used for a variety of functions, including pulse output with an arbitrary duty cycle.

### 12.1.1 Features

The features of the 8-bit timer module are listed below.

- Selection of four clock sources
  The counters can be driven by one of three internal clock signals (ø/8, ø/64, or ø/8192) or an external clock input (enabling use as an external event counter).
- Selection of three ways to clear the counters
  The counters can be cleared on compare match A or B, or by an external reset signal.
- Timer output control by a combination of two compare match signals
  The timer output signal in each channel is controlled by a combination of two independent compare match signals, enabling the timer to generate output waveforms with an arbitrary duty cycle or PWM output.
- Provision for cascading of two channels
  - Operation as a 16-bit timer is possible, using channel 0 for the upper 8 bits and channel 1 for the lower 8 bits (16-bit count mode).
  - Channel 1 can be used to count channel 0 compare matches (compare match count mode).
- Three independent interrupts
  Compare match A and B and overflow interrupts can be requested independently.

### 12.1.2 Block Diagram

Figure 12-1 shows a block diagram of the 8-bit timer module.

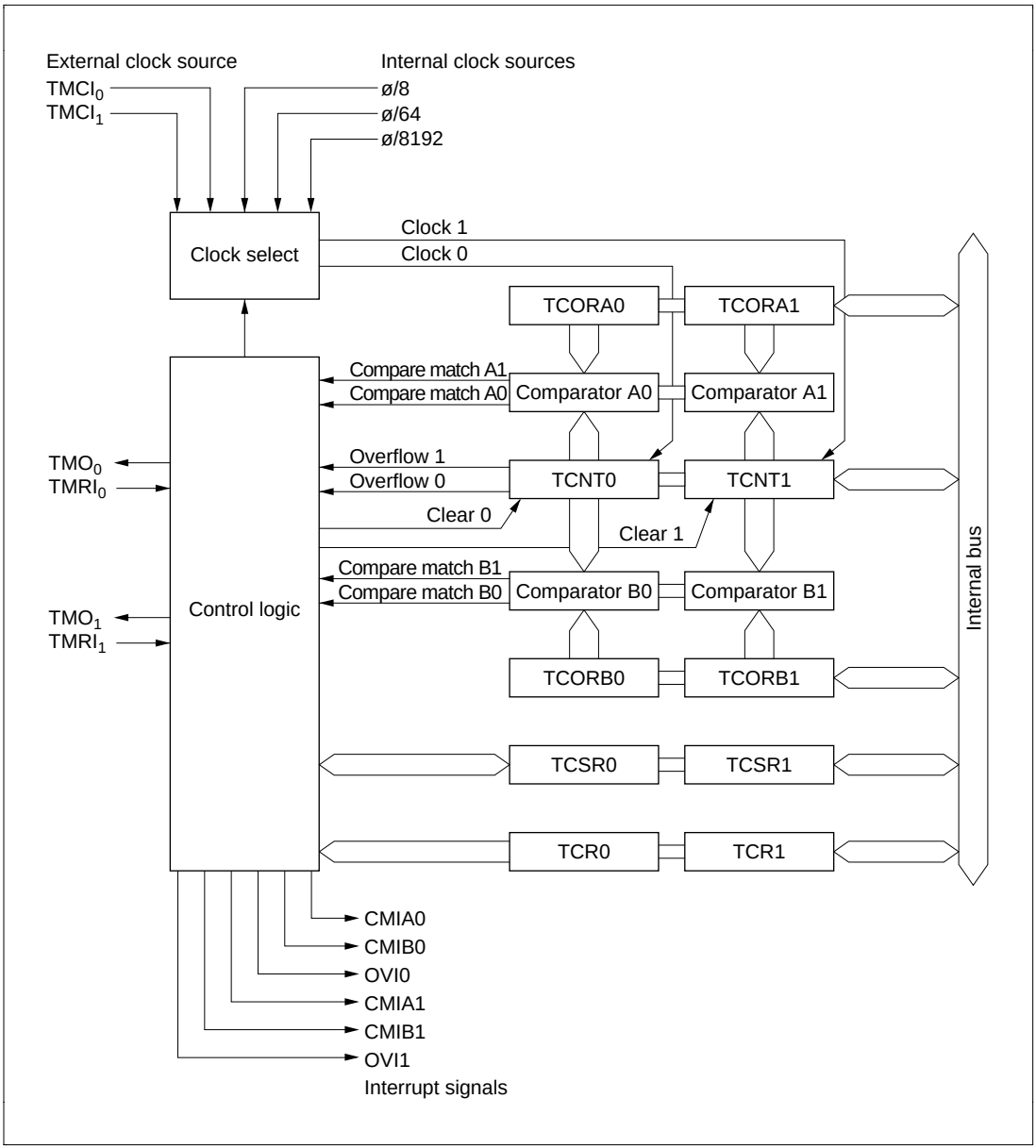



**Figure 12-1 Block Diagram of 8-Bit Timer**

### 12.1.3 Pin Configuration

Table 12-1 summarizes the input and output pins of the 8-bit timer.

**Table 12-1 Input and Output Pins of 8-Bit Timer**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Timer output pin 0 | $TMO_0$ | Output | Outputs at compare match |
| | Timer clock input pin 0 | $TMCI_0$ | Input | Inputs external clock for counter |
| | Timer reset input pin 0 | $TMRI_0$ | Input | Inputs external reset to counter |
| 1 | Timer output pin 1 | $TMO_1$ | Output | Outputs at compare match |
| | Timer clock input pin 1 | $TMCI_1$ | Input | Inputs external clock for counter |
| | Timer reset input pin 1 | $TMRI_1$ | Input | Inputs external reset to counter |

### 12.1.4 Register Configuration

Table 12-2 summarizes the registers of the 8-bit timer module.

**Table 12-2 8-Bit Timer Registers**

| Channel | Name | Abbreviation | R/W | Initial value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Timer control register 0 | TCR0 | R/W | H'00 | H'FFB0 |
| | Timer control/status register 0 | TCSR0 | R/(W)*[2] | H'00 | H'FFB2 |
| | Time constant register A0 | TCORA0 | R/W | H'FF | H'FFB4 |
| | Time constant register B0 | TCORB0 | R/W | H'FF | H'FFB6 |
| | Timer counter 0 | TCNT0 | R/W | H'00 | H'FFB8 |
| 1 | Timer control register 1 | TCR1 | R/W | H'00 | H'FFB1 |
| | Timer control/status register 1 | TCSR1 | R/(W)*[2] | H'10 | H'FFB3 |
| | Time constant register A1 | TCORA1 | R/W | H'FF | H'FFB5 |
| | Time constant register B1 | TCORB1 | R/W | H'FF | H'FFB7 |
| | Timer counter 1 | TCNT1 | R/W | H'00 | H'FFB9 |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address
2. Only 0 can be written to bits 7 to 5, to clear these flags.

Each pair of registers for channel 0 and channel 1 is a 16-bit register with the upper 8 bits for channel 0 and the lower 8 bits for channel 1, so they can be accessed together by word transfer instruction.

## 12.2 Register Descriptions

### 12.2.1 Timer Counters 0 and 1 (TCNT0, TCNT1)

| | | TCNT0 | | | | | | | | TCNT1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCNT0 and TCNT1 are 8-bit readable/writable up-counters that increment on pulses generated from an internal or external clock source. This clock source is selected by clock select bits CKS2 to CKS0 of TCR. The CPU can read or write to TCNT0 and TCNT1 at all times.

TCNT0 and TCNT1 comprise a single 16-bit register, so they can be accessed together by word transfer instruction.

TCNT0 and TCNT1 can be cleared by an external reset input or by a compare match signal. Which signal is to be used for clearing is selected by clock clear bits CCLR1 and CCLR0 of TCR.

When a timer counter overflows from H'FF to H'00, OVF in TCSR is set to 1.

TCNT0 and TCNT1 are each initialized to H'00 by a reset and in hardware standby mode.

### 12.2.2 Time Constant Registers A0 and A1 (TCORA0, TCORA1)

| | | TCORA0 | | | | | | | | TCORA1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value: | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCORA0 and TCORA1 are 8-bit readable/writable registers. TCORA0 and TCORA1 comprise a single 16-bit register so they can be accessed together by word transfer instruction.

TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag of TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of bits OS1 and OS0 of TCSR.

TCORA0 and TCORA1 are each initialized to H'FF by a reset and in hardware standby mode.

### 12.2.3 Time Constant Registers B0 and B1 (TCORB0, TCORB1)

| | | TCORB0 | | | | | | | | TCORB1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value: | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCORB0 and TCORB1 are 8-bit readable/writable registers. TCORB0 and TCORB1 comprise a single 16-bit register so they can be accessed together by word transfer instruction.

TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding CMFB flag of TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of output select bits OS3 and OS2 of TCSR.

TCORB0 and TCORB1 are each initialized to H'FF by a reset and in hardware standby mode.

### 12.2.4 Time Control Registers 0 and 1 (TCR0, TCR1)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCR0 and TCR1 are 8-bit readable/writable registers that select the clock source and the time at which TCNT is cleared, and enable interrupts.

TCR0 and TCR1 are each initialized to H'00 by a reset and in hardware standby mode.

For details of this timing, see section 12.3, Operation.

**Bit 7—Compare Match Interrupt Enable B (CMIEB):** Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag of TCSR is set to 1.

**Bit 7**

| CMIEB | Description | |
|---|---|---|
| 0 | CMFB interrupt requests (CMIB) are disabled | (Initial value) |
| 1 | CMFB interrupt requests (CMIB) are enabled | |

**Bit 6—Compare Match Interrupt Enable A (CMIEA):** Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag of TCSR is set to 1.

**Bit 6**

| CMIEA | Description | |
|---|---|---|
| 0 | CMFA interrupt requests (CMIA) are disabled | (Initial value) |
| 1 | CMFA interrupt requests (CMIA) are enabled | |

**Bit 5—Timer Overflow Interrupt Enable (OVIE):** Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag of TCSR is set to 1.

**Bit 5**

| OVIE | Description | |
|---|---|---|
| 0 | OVF interrupt requests (OVI) are disabled | (Initial value) |
| 1 | OVF interrupt requests (OVI) are enabled | |

**Bits 4 and 3—Counter Clear 1 and 0 (CCLR1 and CCLR0):** These bits select the method by which TCNT is cleared: by compare match A or B, or by an external reset input.

| Bit 4 | Bit 3 | | |
|---|---|---|---|
| CCLR1 | CCLR0 | Description | |
| 0 | 0 | Clear is disabled | (Initial value) |
| | 1 | Clear by compare match A | |
| 1 | 0 | Clear by compare match B | |
| | 1 | Clear by rising edge of external reset input | |

**Bits 2 to 0—Clock Select 2 to 0 (CKS2 to CKS0):** These bits select whether the clock input to TCNT is an internal or external clock.

Three internal clocks can be selected, all divided from the system clock (ø): ø/8, ø/64, and ø/8192. The falling edge of the selected internal clock triggers the count.

When use of an external clock is selected, three types of count can be selected: at the rising edge, the falling edge, and both rising and falling edges.

Some functions differ between channel 0 and channel 1.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| **CKS2** | **CKS1** | **CKS0** | **Description** | |
| 0 | 0 | 0 | Clock input disabled | (Initial value) |
| | | 1 | Internal clock, counted at falling edge of ø/8 | |
| | 1 | 0 | Internal clock, counted at falling edge of ø/64 | |
| | | 1 | Internal clock, counted at falling edge of ø/8192 | |
| 1 | 0 | 0 | For channel 0: count at TCNT1 overflow signal* | |
| | | | For channel 1: count at TCNT0 compare match A* | |
| | | 1 | External clock, counted at rising edge | |
| | 1 | 0 | External clock, counted at falling edge | |
| | | 1 | External clock, counted at both rising and falling edges | |

Note: * If the count input of channel 0 is the TCNT1 overflow signal and that of channel 1 is the TCNT0 compare match signal, no incrementing clock is generated. Do not use this setting.

### 12.2.5 Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1)

**TCSR0**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/W | R/W | R/W | R/W | R/W |

**TCSR1**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 |
| Initial value | : | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | — | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to bits 7 to 5, to clear these flags.

TCSR0 and TCSR1 are 8-bit registers that display compare match and overflow statuses, and control compare match output.

TCSR0 is initialized to H'00, and TCSR1 to H'10, by a reset and in hardware standby mode.

**Bit 7—Compare Match Flag B (CMFB):** Status flag indicating whether the values of TCNT and TCORB match.

**Bit 7**

| CMFB | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB<br>• When DTC is activated by CMIB interrupt while DISEL bit of MRB in DTC is 0 |
| 1 | [Setting condition]<br>Set when TCNT matches TCORB |

**Bit 6—Compare Match Flag A (CMFA):** Status flag indicating whether the values of TCNT and TCORA match.

**Bit 6**

| CMFA | Description | |
|---|---|---|
| 0 | [Clearing conditions] | (Initial value) |
| | • Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA | |
| | • When DTC is activated by CMIA interrupt while DISEL bit of MRB in DTC is 0 | |
| 1 | [Setting condition] | |
| | Set when TCNT matches TCORA | |

**Bit 5—Timer Overflow Flag (OVF):** Status flag indicating that TCNT has overflowed (changed from H'FF to H'00).

**Bit 5**

| OVF | Description | |
|---|---|---|
| 0 | [Clearing condition] | (Initial value) |
| | Cleared by reading OVF when OVF = 1, then writing 0 to OVF | |
| 1 | [Setting condition] | |
| | Set when TCNT overflows from H'FF to H'00 | |

**Bit 4—A/D Trigger Enable (ADTE) (TCSR0 Only):** Selects enabling or disabling of A/D converter start requests by compare-match A.

In TCSR1, this bit is reserved: it is always read as 1 and cannot be modified.

**Bit 4**

| ADTE | Description | |
|---|---|---|
| 0 | A/D converter start requests by compare match A are disabled | (Initial value) |
| 1 | A/D converter start requests by compare match A are enabled | |

**Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0):** These bits specify how the timer output level is to be changed by a compare match of TCOR and TCNT.

Bits OS3 and OS2 select the effect of compare match B on the output level, bits OS1 and OS0 select the effect of compare match A on the output level, and both of them can be controlled independently.

Note, however, that priorities are set such that: toggle output > 1 output > 0 output. If compare matches occur simultaneously, the output changes according to the compare match with the higher priority.

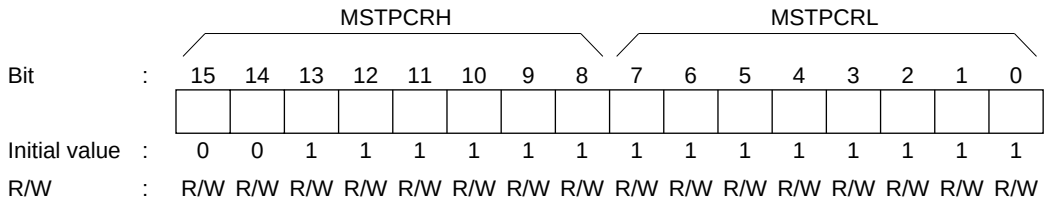
Timer output is disabled when bits OS3 to OS0 are all 0.

After a reset, the timer output is 0 until the first compare match event occurs.

| Bit 3 | Bit 2 | | |
|---|---|---|---|
| **OS3** | **OS2** | **Description** | |
| 0 | 0 | No change when compare match B occurs | (Initial value) |
| | 1 | 0 is output when compare match B occurs | |
| 1 | 0 | 1 is output when compare match B occurs | |
| | 1 | Output is inverted when compare match B occurs (toggle output) | |

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **OS1** | **OS0** | **Description** | |
| 0 | 0 | No change when compare match A occurs | (Initial value) |
| | 1 | 0 is output when compare match A occurs | |
| 1 | 0 | 1 is output when compare match A occurs | |
| | 1 | Output is inverted when compare match A occurs (toggle output) | |

### 12.2.6 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP12 bit in MSTPCR is set to 1, the 8-bit timer operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 12—Module Stop (MSTP12):** Specifies the 8-bit timer stop mode.

**Bit 12**

| MSTP12 | Description | |
|---|---|---|
| 0 | 8-bit timer module stop mode cleared | |
| 1 | 8-bit timer module stop mode set | (Initial value) |

## 12.3 Operation

### 12.3.1 TCNT Incrementation Timing

TCNT is incremented by input clock pulses (either internal or external).

**Internal Clock:** Three different internal clock signals (ø/8, ø/64, or ø/8192) divided from the system clock (ø) can be selected, by setting bits CKS2 to CKS0 in TCR. Figure 12-2 shows the count timing.

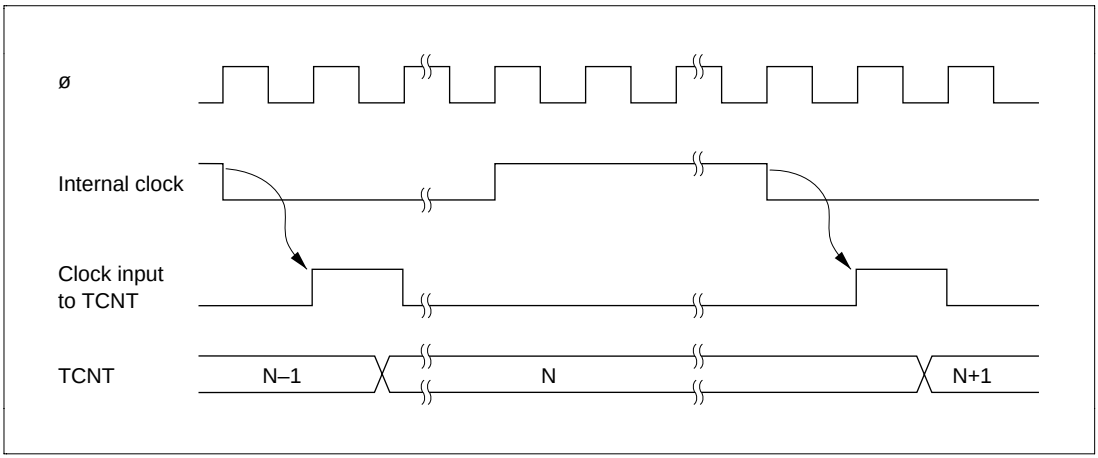



**Figure 12-2 Count Timing for Internal Clock Input**

**External Clock:** Three incrementation methods can be selected by setting bits CKS2 to CKS0 in TCR: at the rising edge, the falling edge, and both rising and falling edges.

Note that the external clock pulse width must be at least 1.5 states for incrementation at a single edge, and at least 2.5 states for incrementation at both edges. The counter will not increment correctly if the pulse width is less than these values.

Figure 12-3 shows the timing of incrementation at both edges of an external clock signal.

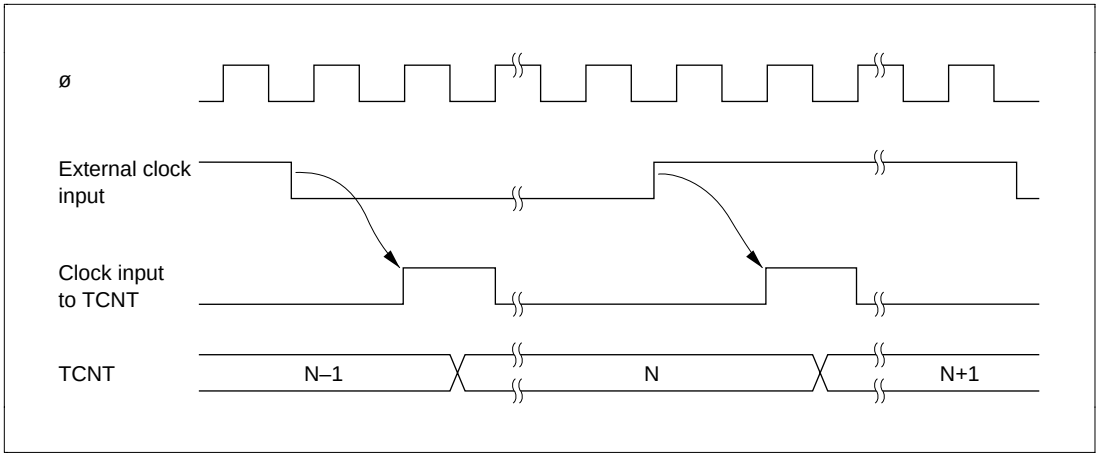



**Figure 12-3 Count Timing for External Clock Input**

### 12.3.2 Compare Match Timing

**Setting of Compare Match Flags A and B (CMFA, CMFB):** The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated.

Therefore, when TCOR and TCNT match, the compare match signal is not generated until the next incrementation clock input. Figure 12-4 shows this timing.

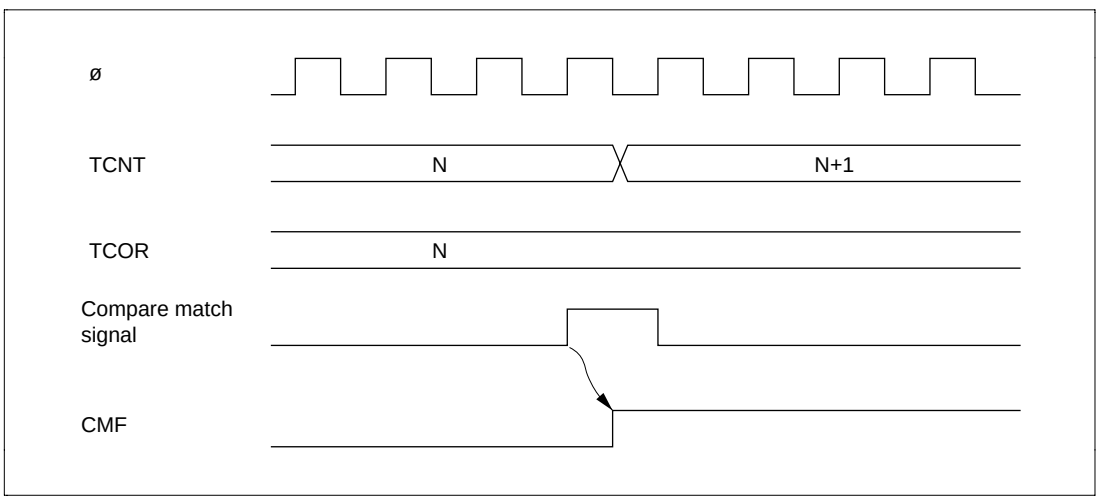



**Figure 12-4 Timing of CMF Setting**

**Timer Output Timing:** When compare match A or B occurs, the timer output changes a specified by bits OS3 to OS0 in TCSR. Depending on these bits, the output can remain the same, change to 0, change to 1, or toggle.

Figure 12-5 shows the timing when the output is set to toggle at compare match A.

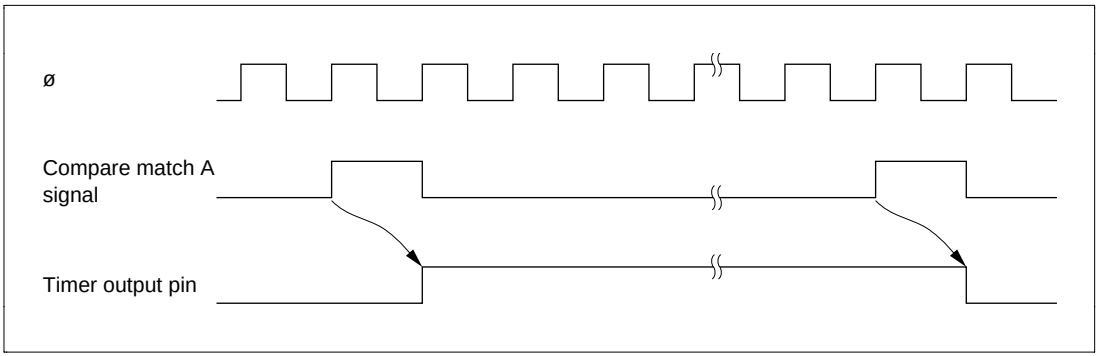



**Figure 12-5 Timing of Timer Output**

**Timing of Compare Match Clear:** The timer counter is cleared when compare match A or B occurs, depending on the setting of the CCLR1 and CCLR0 bits in TCR. Figure 12-6 shows the timing of this operation.

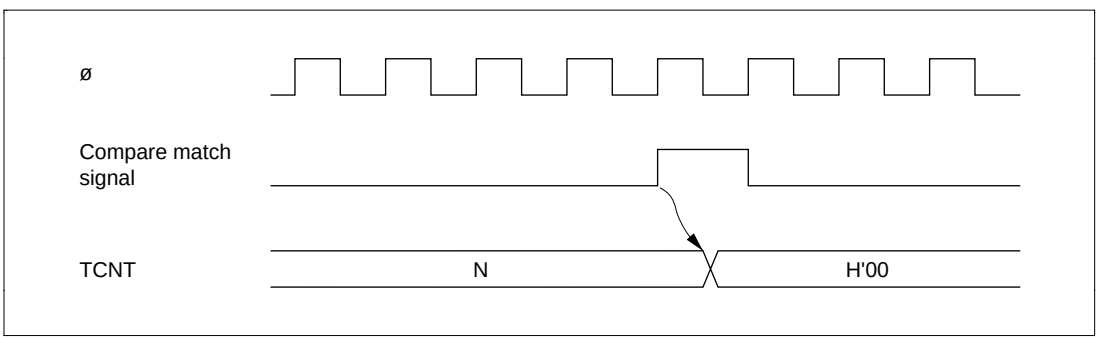



**Figure 12-6 Timing of Compare Match Clear**

### 12.3.3 Timing of External RESET on TCNT

TCNT is cleared at the rising edge of an external reset input, depending on the settings of the CCLR1 and CCLR0 bits in TCR. The clear pulse width must be at least 1.5 states. Figure 12-7 shows the timing of this operation.

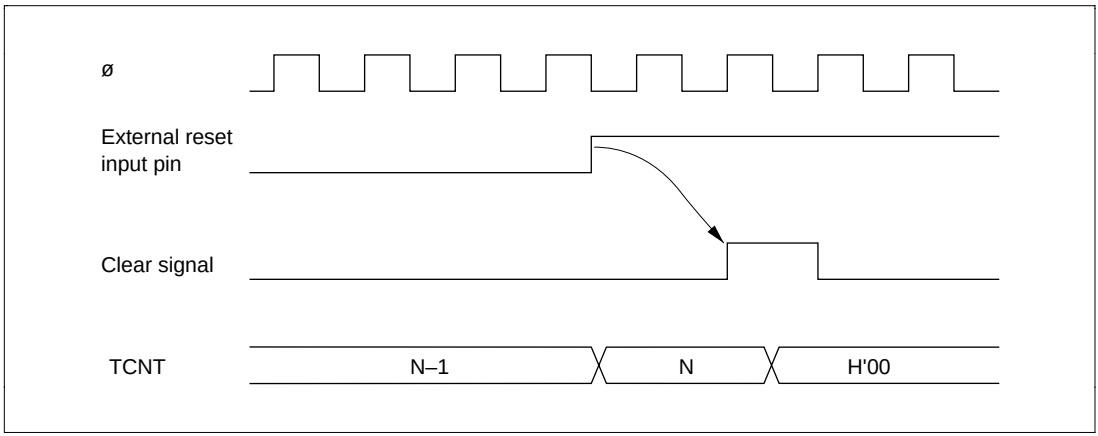



**Figure 12-7 Timing of External Reset**

### 12.3.4 Timing of Overflow Flag (OVF) Setting

The OVF in TCSR is set to 1 when the timer count overflows (changes from H'FF to H'00). Figure 12-8 shows the timing of this operation.

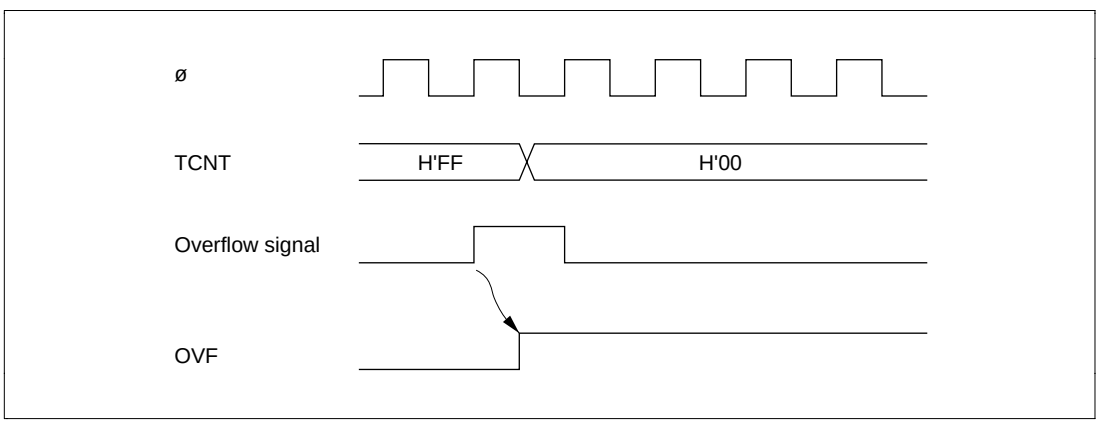



**Figure 12-8 Timing of OVF Setting**

### 12.3.5 Operation with Cascaded Connection

If bits CKS2 to CKS0 in either TCR0 or TCR1 are set to B’100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer could be used (16-bit timer mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match counter mode). In this case, the timer operates as below.

**16-Bit Counter Mode:** When bits CKS2 to CKS0 in TCR0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

- Setting of compare match flags
  - The CMF flag in TCSR0 is set to 1 when a 16-bit compare match event occurs.
  - The CMF flag in TCSR1 is set to 1 when a lower 8-bit compare match event occurs.

- Counter clear specification
  - If the CCLR1 and CCLR0 bits in TCR0 have been set for counter clear at compare match, the 16-bit counter (TCNT0 and TCNT1 together) is cleared when a 16-bit compare match event occurs. The 16-bit counter (TCNT0 and TCNT1 together) is cleared even if counter clear by the $TMRI_0$ pin has also been set.
  - The settings of the CCLR1 and CCLR0 bits in TCR1 are ignored. The lower 8 bits cannot be cleared independently.

- Pin output
  - Control of output from the $TMO_0$ pin by bits OS3 to OS0 in TCSR0 is in accordance with the 16-bit compare match conditions.
  - Control of output from the $TMO_1$ pin by bits OS3 to OS0 in TCSR1 is in accordance with the lower 8-bit compare match conditions.

**Compare Match Counter Mode:** When bits CKS2 to CKS0 in TCR1 are B'100, TCNT1 counts compare match A’s for channel 0.

Channels 1 and 0 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clear are in accordance with the settings for each channel.

**Note on Usage:** If the 16-bit counter mode and compare match counter mode are set simultaneously, the input clock pulses for TCNT0 and TCNT1 are not generated and thus the counters will stop operating. Software should therefore avoid using both these modes.

## 12.4 Interrupt Sources

There are three 8-bit timer interrupt sources: CMIA, CMIB, and OVI. Their relative priorities are shown in Table 12-3. Each interrupt source is set as enabled or disabled by the corresponding interrupt enable bit in TCR, and independent interrupt requests are sent for each to the interrupt controller. It is also possible to activate the DTC by means of CMIA and CMIB interrupts.

**Table 12-3 8-Bit Timer Interrupt Sources**

| Interrupt Source | Description | DTC Activation | Priority |
|---|---|---|---|
| CMIA | Interrupt by CMFA | Possible | High |
| CMIB | Interrupt by CMFB | Possible | ↑ |
| OVI | Interrupt by OVF | Not possible | Low |

## 12.5 Sample Application

In the example below, the 8-bit timer is used to generate a pulse output with a selected duty cycle, as shown in figure 12-9. The control bits are set as follows:

[1] In TCR, bit CCLR1 is cleared to 0 and bit CCLR0 is set to 1 so that the timer counter is cleared when its value matches the constant in TCORA.

[2] In TCSR, bits OS3 to OS0 are set to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.

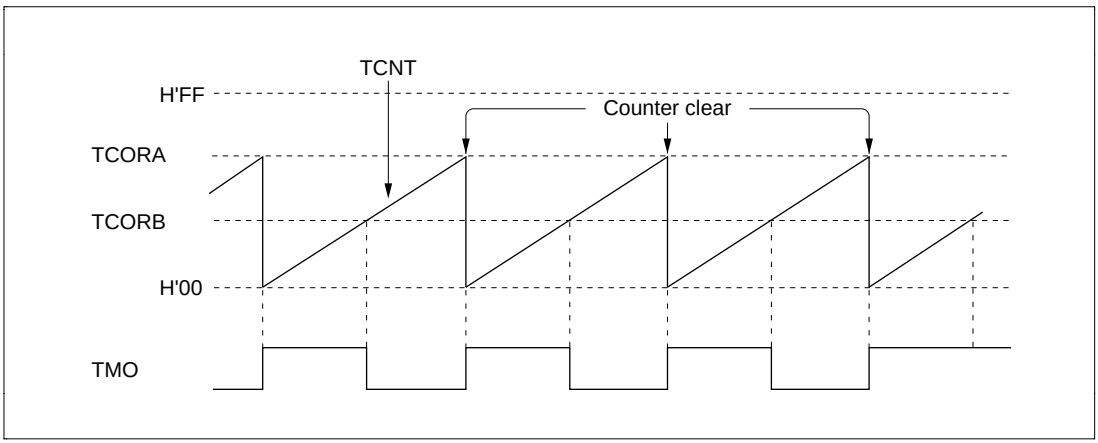



**Figure 12-9 Example of Pulse Output**

## 12.6 Usage Notes

Application programmers should note that the following kinds of contention can occur in the 8-bit timer.

### 12.6.1 Contention between TCNT Write and Clear

If a timer counter clock pulse is generated during the T2 state of a TCNT write cycle, the clear takes priority, so that the counter is cleared and the write is not performed.

Figure 12-10 shows this operation.

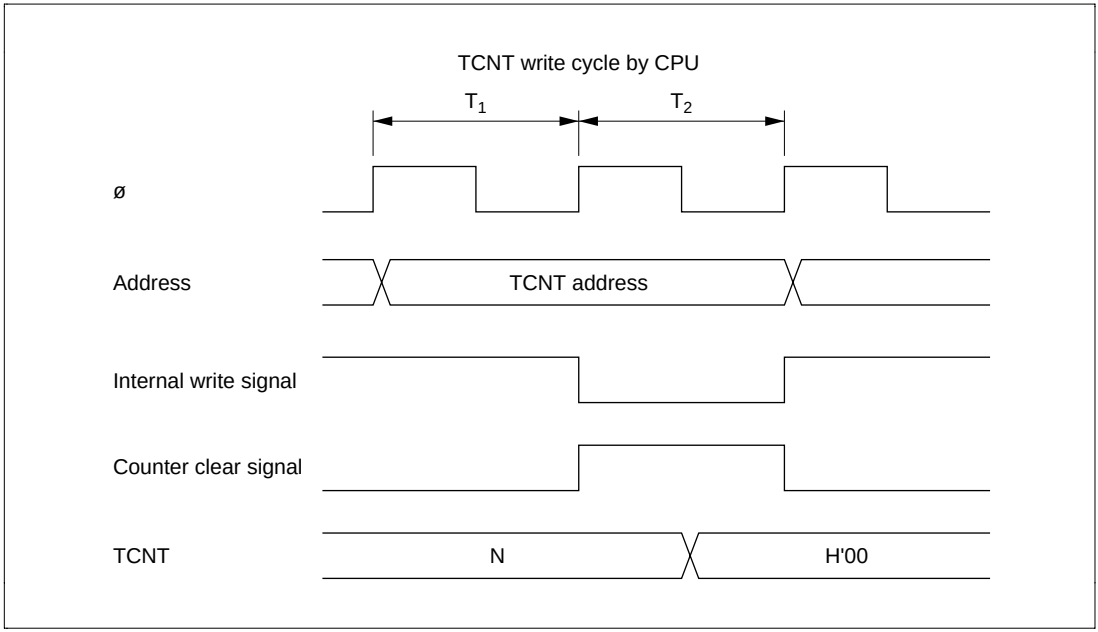



**Figure 12-10 Contention between TCNT Write and Clear**

### 12.6.2 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the counter is not incremented.

Figure 12-11 shows this operation.

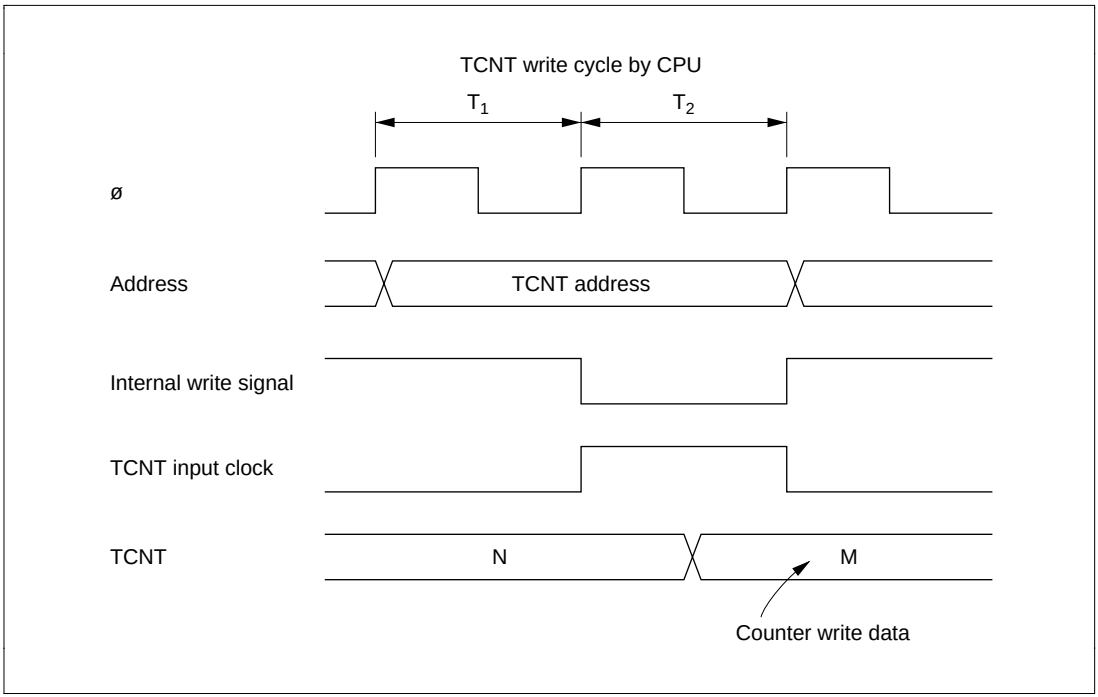



**Figure 12-11 Contention between TCNT Write and Increment**

### 12.6.3 Contention between TCOR Write and Compare Match

During the T2 state of a TCOR write cycle, the TCOR write has priority even if a compare match event occurs.

Figure 12-12 shows this operation.

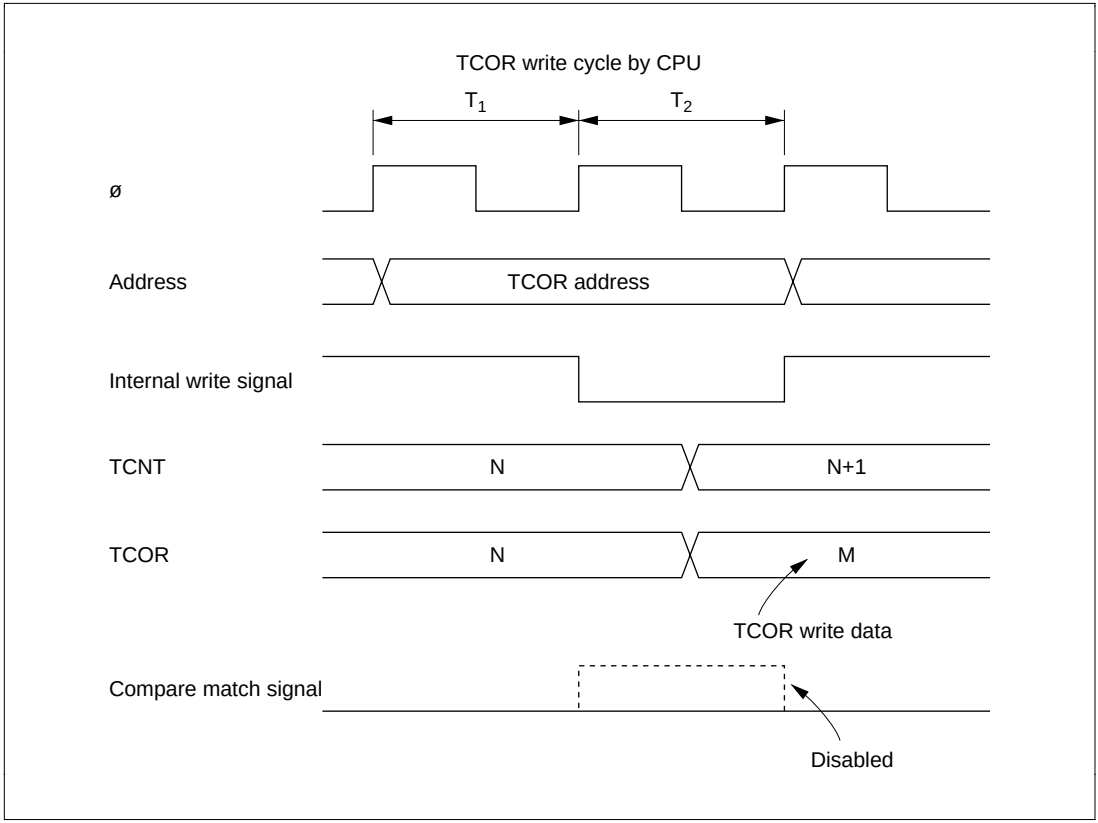



**Figure 12-12 Contention between TCOR Write and Compare Match**

### 12.6.4 Contention between Compare Matches A and B

If compare match events A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output statuses set for compare match A and compare match B, as shown in table 12-4.

**Table 12-4 Timer Output Priorities**

| Output Setting | Priority |
|---|---|
| Toggle output | High |
| 1 output | ↑ |
| 0 output | |
| No change | Low |

### 12.6.5 Switching of Internal Clocks and TCNT Operation

TCNT may increment erroneously when the internal clock is switched over. Table 12-5 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the falling edge of the internal clock pulse is detected. If clock switching causes a change from high to low level, as shown in case 3 in table 12-5, a TCNT clock pulse is generated on the assumption that the switchover is a falling edge. This increments TCNT.

The erroneous incrementation can also happen when switching between internal and external clocks.

**Table 12-5 Switching of Internal Clock and TCNT Operation**

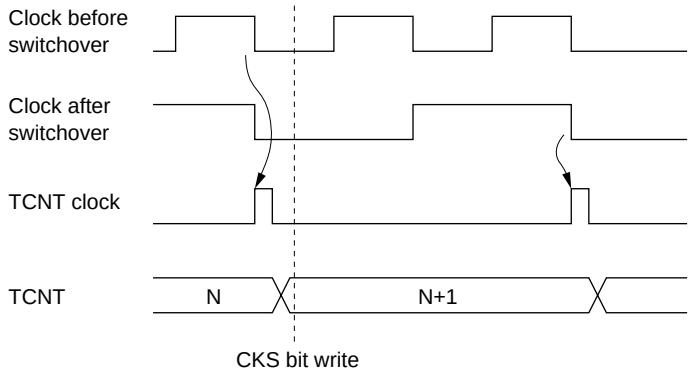
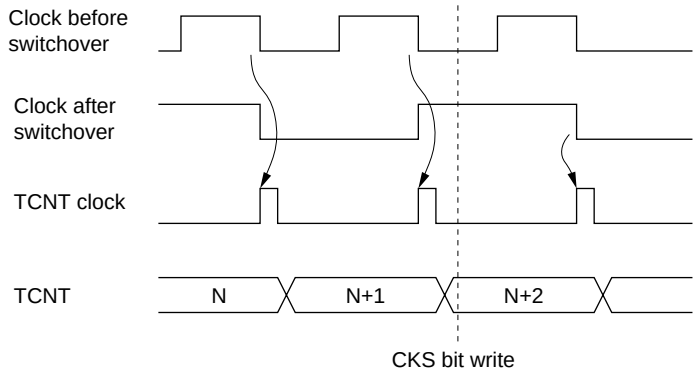
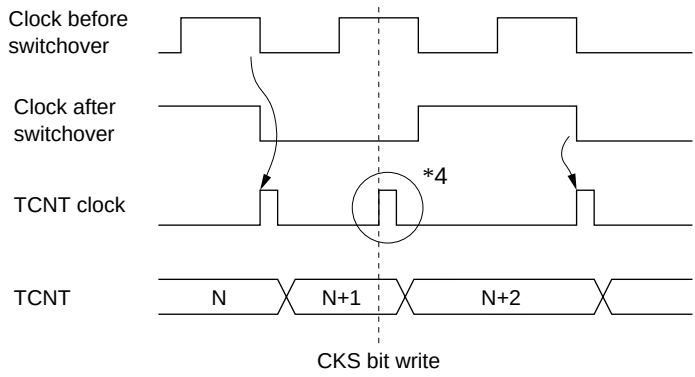
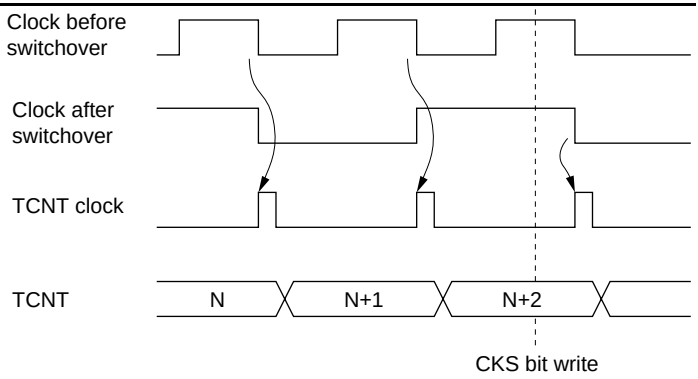
| No. | Timing of Switchover by Means of CKS1 and CKS0 Bits | TCNT Clock Operation |
|---|---|---|
| 1 | Switching from low to low*1 |  |
| 2 | Switching from low to high*2 |  |
| 3 | Switching from high to low*3 |  |

**Table 12-5 Switching of Internal Clock and TCNT Operation (cont)**

| No. | Timing of Switchover by Means of CKS1 and CKS0 Bits | TCNT Clock Operation |
|---|---|---|
| 4 | Switching from high to high |  |



Notes: 1. Includes switching from low to stop, and from stop to low.
2. Includes switching from stop to high.
3. Includes switching from high to stop.
4. Generated on the assumption that the switchover is a falling edge; TCNT is incremented.

# Section 13 Watchdog Timer

## 13.1 Overview

The H8S/2655 Series has a single-channel on-chip watchdog timer (WDT) for monitoring system operation. The WDT outputs an overflow signal ($\overline{\text{WDTOVF}}$) if a system crash prevents the CPU from writing to the timer counter, allowing it to overflow. At the same time, the WDT can also generate an internal reset signal for the H8S/2655 Series.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

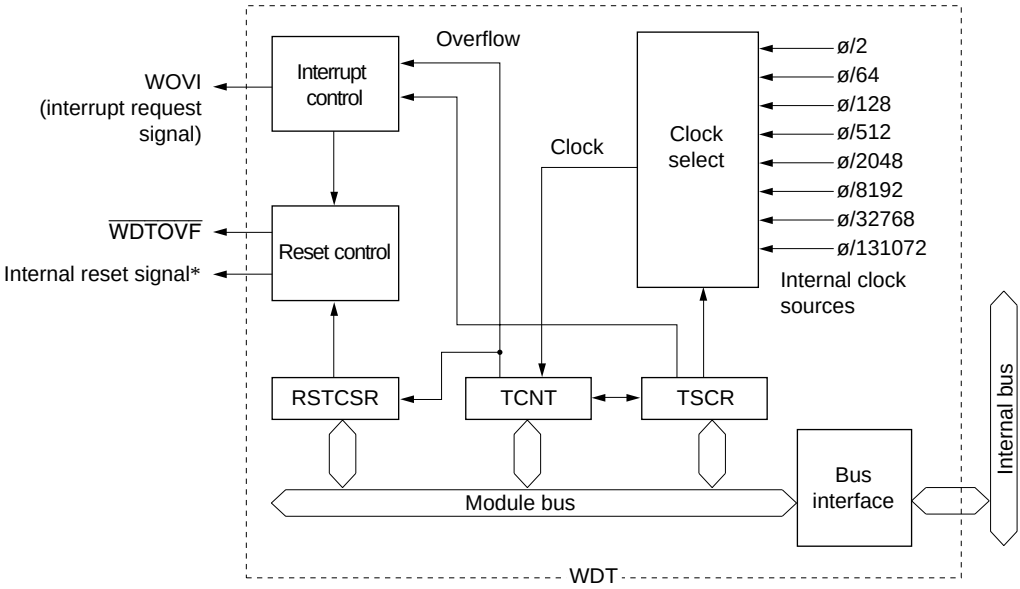
### 13.1.1 Features

WDT features are listed below.

- Switchable between watchdog timer mode and interval timer mode
- $\overline{\text{WDTOVF}}$ output when in watchdog timer mode
  If the counter overflows, the WDT outputs $\overline{\text{WDTOVF}}$. It is possible to select whether or not the entire H8S/2655 Series is reset at the same time. This internal reset can be a power-on reset or a manual reset.
- Interrupt generation when in interval timer mode
  If the counter overflows, the WDT generates an interval timer interrupt.
- Choice of eight counter clock sources.

### 13.1.2 Block Diagram

Figure 13-1 shows a block diagram of the WDT.




**Legend**
TCSR : Timer control/status register
TCNT : Timer counter
RSTCSR : Reset control/status register

Note: * The type of internal reset signal depends on a register setting. Either power-on reset or manual reset can be selected.

**Figure 13-1 Block Diagram of WDT**

### 13.1.3 Pin Configuration

Table 13-1 describes the WDT output pin.

**Table 13-1 WDT Pin**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Watchdog timer overflow | $\overline{\text{WDTOVF}}$ | Output | Outputs counter overflow signal in watchdog timer mode |

### 13.1.4 Register Configuration

The WDT has three registers, as summarized in table 13-2. These registers control clock selection, WDT mode switching, and the reset signal.

**Table 13-2 WDT Registers**

| | | | | Address*[1] | |
|---|---|---|---|---|---|
| Name | Abbreviation | R/W | Initial Value | Write*[2] | Read |
| Timer control/status register | TCSR | R/(W)*[3] | H'18 | H'FFBC | H'FFBC |
| Timer counter | TCNT | R/W | H'00 | H'FFBC | H'FFBD |
| Reset control/status register | RSTCSR | R/(W)*[3] | H'1F | H'FFBE | H'FFBF |

Notes: 1. Lower 16 bits of the address.
2. For details of write operations, see section 13.2.4, Notes on Register Access.
3. Only a write of 0 is permitted to bit 7, to clear the flag.

## 13.2 Register Descriptions

### 13.2.1 Timer Counter (TCNT)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCNT is an 8-bit readable/writable* up-counter.

When the TME bit is set to 1 in TCSR, TCNT starts counting pulses generated from the internal clock source selected by bits CKS2 to CKS0 in TCSR. When the count overflows (changes from H'FF to H'00), either the watchdog timer overflow signal ($\overline{\text{WDTOVF}}$) or an interval timer interrupt (WOVI) is generated, depending on the mode selected by the WT/$\overline{\text{IT}}$ bit in TCSR.

TCNT is initialized to H'00 by a reset, in hardware standby mode, or when the TME bit is cleared to 0. It is not initialized in software standby mode.

Note: * The method for writing to TCNT is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

### 13.2.2 Timer Control/Status Register (TCSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | OVF | WT/$\overline{\text{IT}}$ | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | — | — | R/W | R/W | R/W |

Note: * Can only be written with 0 for flag clearing.

TCSR is an 8-bit readable/writable* register. Its functions include selecting the clock source to be input to TCNT, and the timer mode.

TCR is initialized to H'18 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Note: * The method for writing to TCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

**Bit 7—Overflow Flag (OVF):** Indicates that TCNT has overflowed from H'FF to H'00, when in interval timer mode. This flag cannot be set during watchdog timer operation.

**Bit 7**

| OVF | Description | |
|---|---|---|
| 0 | [Clearing condition] | |
| | Cleared by reading TCSR when OVF = 1, then writing 0 to OVF | (Initial value) |
| 1 | [Setting condition] | |
| | Set when TCNT overflows (changes from H'FF to H'00) in interval timer mode | |

**Bit 6—Timer Mode Select (WT/$\overline{IT}$):** Selects whether the WDT is used as a watchdog timer or interval timer. If used as an interval timer, the WDT generates an interval timer interrupt request (WOVI) when TCNT overflows. If used as a watchdog timer, the WDT generates the $\overline{WDTOVF}$ signal when TCNT overflows.

**Bit 6**

| WT/$\overline{IT}$ | Description | |
|---|---|---|
| 0 | Interval timer: Sends the CPU an interval timer interrupt request (WOVI) when TCNT overflows | (Initial value) |
| 1 | Watchdog timer: Generates the $\overline{WDTOVF}$ signal when TCNT overflows | |

Note: * For details of the case where TCNT overflows in watchdog timer mode, see section 13.2.3, Reset Control/Status Register (RSTCSR).

**Bit 5—Timer Enable (TME):** Selects whether TCNT runs or is halted.

**Bit 5**

| TME | Description | |
|---|---|---|
| 0 | TCNT is initialized to H'00 and halted | (Initial value) |
| 1 | TCNT counts | |

**Bits 4 and 3—Reserved:** Read-only bits, always read as 1.

**Bits 2 to 0:** Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of eight internal clock sources, obtained by dividing the system clock (ø), for input to TCNT.

| Bit 2 | Bit 1 | Bit 0 | Description | |
|---|---|---|---|---|
| **CKS2** | **CKS1** | **CKS0** | **Clock** | **Overflow Period (when ø = 20 MHz)*** |
| 0 | 0 | 0 | ø/2 (initial value) | 25.6 µs |
| | | 1 | ø/64 | 819.2 µs |
| | 1 | 0 | ø/128 | 1.6 ms |
| | | 1 | ø/512 | 6.6 ms |
| 1 | 0 | 0 | ø/2048 | 26.2 ms |
| | | 1 | ø/8192 | 104.9 ms |
| | 1 | 0 | ø/32768 | 419.4 ms |
| | | 1 | ø/131072 | 1.68 s |

Note: * The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.

### 13.2.3 Reset Control/Status Register (RSTCSR)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WOVF | RSTE | RSTS | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/(W)* | R/W | R/W | — | — | — | — | — |

Note: * Can only be written with 0 for flag clearing.

RSTCSR is an 8-bit readable/writable* register that controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal.

RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, but not by the WDT internal reset signal caused by overflows.

Note: * The method for writing to RSTCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

**Bit 7—Watchdog Overflow Flag (WOVF):** Indicates that TCNT has overflowed (changed from H'FF to H'00) during watchdog timer operation. This bit is not set in interval timer mode.

**Bit 7**

| WOVF | Description | |
|---|---|---|
| 0 | [Clearing condition] | (Initial value) |
| | Cleared by reading TCSR when WOVF = 1, then writing 0 to WOVF | |
| 1 | [Setting condition] | |
| | Set when TCNT overflows (changed from H'FF to H'00) during watchdog timer operation | |

**Bit 6—Reset Enable (RSTE):** Specifies whether or not a reset signal is generated in the H8S/2655 Series if TCNT overflows during watchdog timer operation.

**Bit 6**

| RSTE | Description | |
|---|---|---|
| 0 | Reset signal is not generated if TCNT overflows* | (Initial value) |
| 1 | Reset signal is generated if TCNT overflows | |

Note: * The modules within the H8S/2655 Series are not reset, but TCNT and TCSR within the WDT are reset.

**Bit 5—Reset Select (RSTS):** Selects the type of internal reset generated if TCNT overflows during watchdog timer operation.

For details of the types of resets, see section 4, Exception Handling.

**Bit 5**

| RSTS | Description | |
|---|---|---|
| 0 | Power-on reset | (Initial value) |
| 1 | Manual reset | |

**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

### 13.2.4 Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

**Writing to TCNT and TCSR:** These registers must be written to by a word transfer instruction. They cannot be written to with byte instructions.

Figure 13-2 shows the format of data written to TCNT and TCSR. TCNT and TCSR both have the same write address. For a write to TCNT, the upper byte of the written word must contain H'5A and the lower byte must contain the write data. For a write to TCSR, the upper byte of the written word must contain H'A5 and the lower byte must contain the write data. This transfers the write data from the lower byte to TCNT or TCSR.

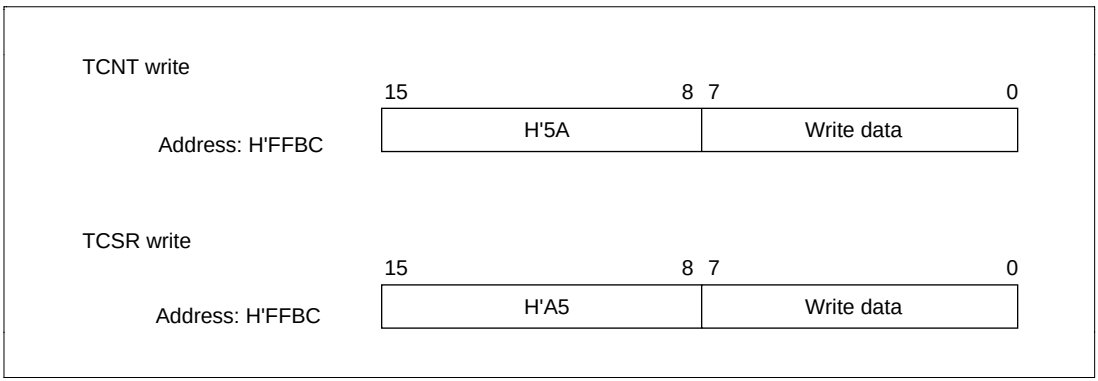



**Figure 13-2 Format of Data Written to TCNT and TCSR**

**Writing to RSTCSR:** RSTCSR must be written to by word transfer instruction to address H'FFBE. It cannot be written to with byte instructions.

Figure 13-3 shows the format of data written to RSTCSR. The method of writing 0 to the WOVF bit differs from that for writing to the RSTE and RSTS bits.

To write 0 to the WOVF bit, the write data must have H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0, but has no effect on the RSTE and RSTS bits. To write to the RSTE and RSTS bits, the upper byte must contain H'5A and the lower byte must contain the write data. This writes the values in bits 6 and 5 of the lower byte into the RSTE and RSTS bits, but has no effect on the WOVF bit.

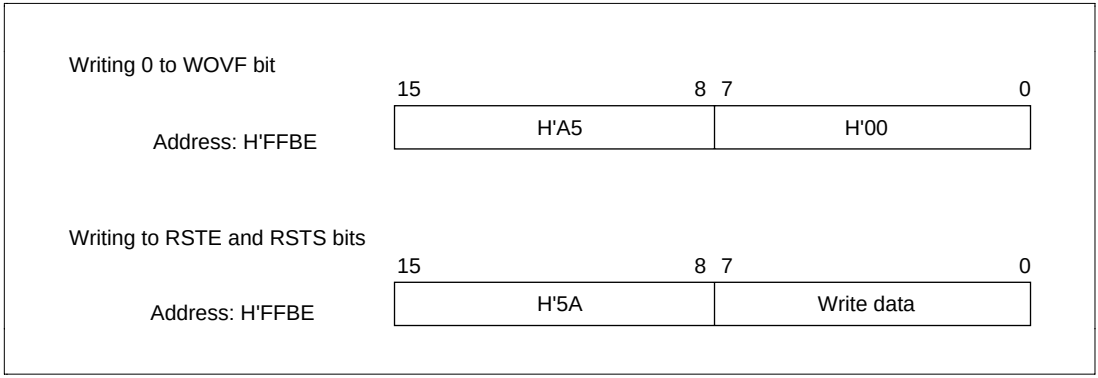



**Figure 13-3 Format of Data Written to RSTCSR**

**Reading TCNT, TCSR, and RSTCSR:** These registers are read in the same way as other registers. The read addresses are H'FFBC for TCSR, H'FFBD for TCNT, and H'FFBF for RSTCSR.

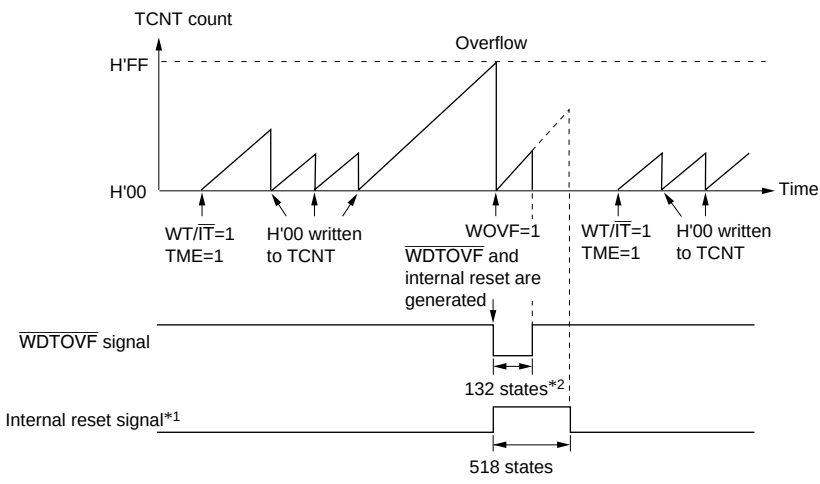
## 13.3 Operation

### 13.3.1 Watchdog Timer Operation

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ and TME bits to 1. Software must prevent TCNT overflows by rewriting the TCNT value (normally be writing H'00) before overflows occurs. This ensures that TCNT does not overflow while the system is operating normally. If TCNT overflows without being rewritten because of a system crash or other error, the $\overline{WDTOVF}$ signal is output. This is shown in figure 13-4. This $\overline{WDTOVF}$ signal can be used to reset the system. The $\overline{WDTOVF}$ signal is output for 132 states when RSTE = 1, and for 130 states when RSTE = 0.

If TCNT overflows when 1 is set in the RSTE bit in RSTCSR, a signal that resets the H8S/2655 Series internally is generated at the same time as the $\overline{WDTOVF}$ signal. This reset can be selected as a power-on reset or a manual reset, depending on the setting of the RSTS bit in RSTCSR. The internal reset signal is output for 518 states.

If a reset caused by a signal input to the $\overline{RES}$ pin occurs at the same time as a reset caused by a WDT overflow, the $\overline{RES}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.




**Legend**
WT/$\overline{IT}$: Timer mode select bit
TME : Timer enable bit

Notes: 1. The internal reset signal is generated only if the RSTE bit is set to 1.
2. 130 states when the RSTE bit is cleared to 0.

**Figure 13-4 Watchdog Timer Operation**

### 13.3.2 Interval Timer Operation

To use the WDT as an interval timer, clear the WT/$\overline{IT}$ bit in TCSR to 0 and set the TME bit to 1. An interval timer interrupt (WOVI) is generated each time TCNT overflows, provided that the WDT is operating as an interval timer, as shown in figure 13-5. This function can be used to generate interrupt requests at regular intervals.

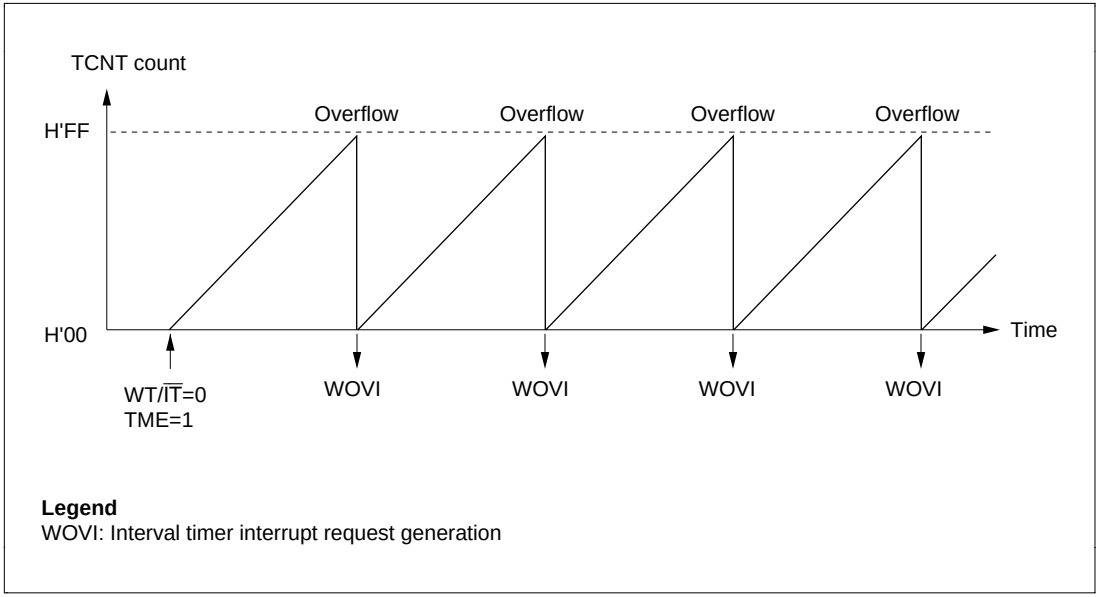



**Figure 13-5 Interval Timer Operation**

### 13.3.3 Timing of Setting Overflow Flag (OVF)

The OVF flag is set to 1 if TCNT overflows during interval timer operation. At the same time, an interval timer interrupt (WOVI) is requested. This timing is shown in figure 13-6.

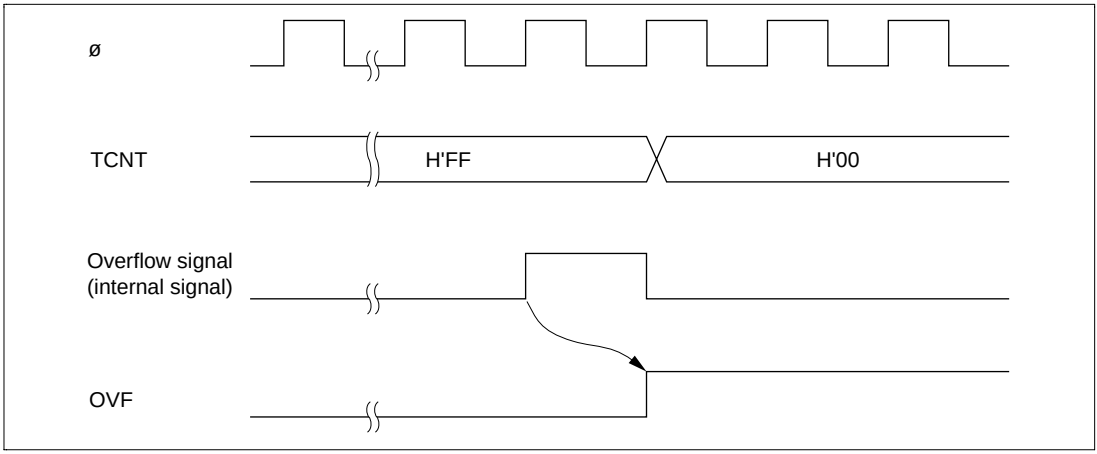



**Figure 13-6 Timing of Setting of OVF**

### 13.3.4 Timing of Setting of Watchdog Timer Overflow Flag (WOVF)

The WOVF flag is set to 1 if TCNT overflows during watchdog timer operation. At the same time, the $\overline{\text{WDTOVF}}$ signal goes low. If TCNT overflows while the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire H8S/2655 Series chip. Figure 13-7 shows the timing in this case.

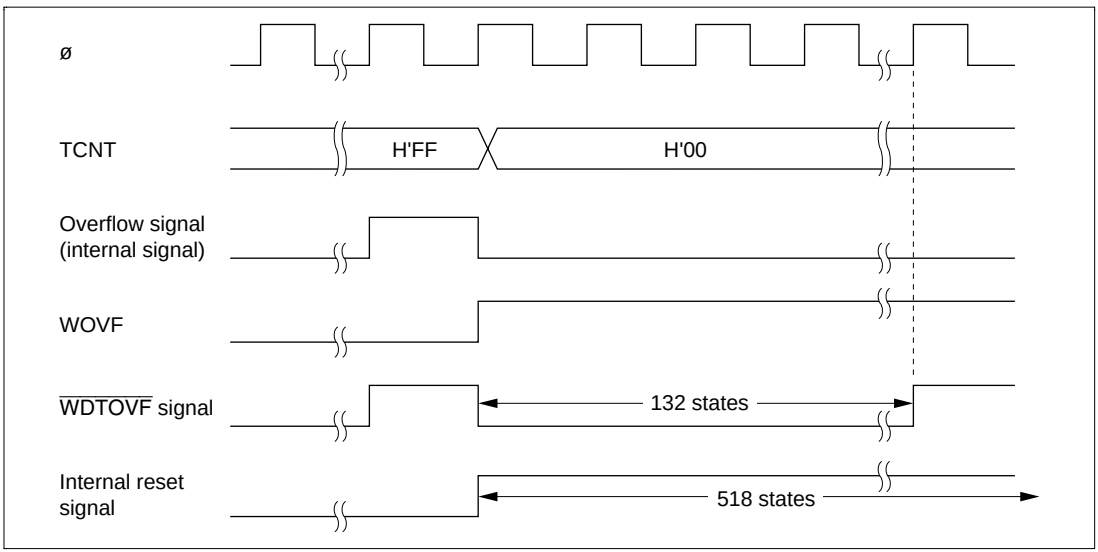



**Figure 13-7 Timing of Setting of WOVF**

## 13.4 Interrupts

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR.

## 13.5 Usage Notes

### 13.5.1 Contention between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 13-8 shows this operation.

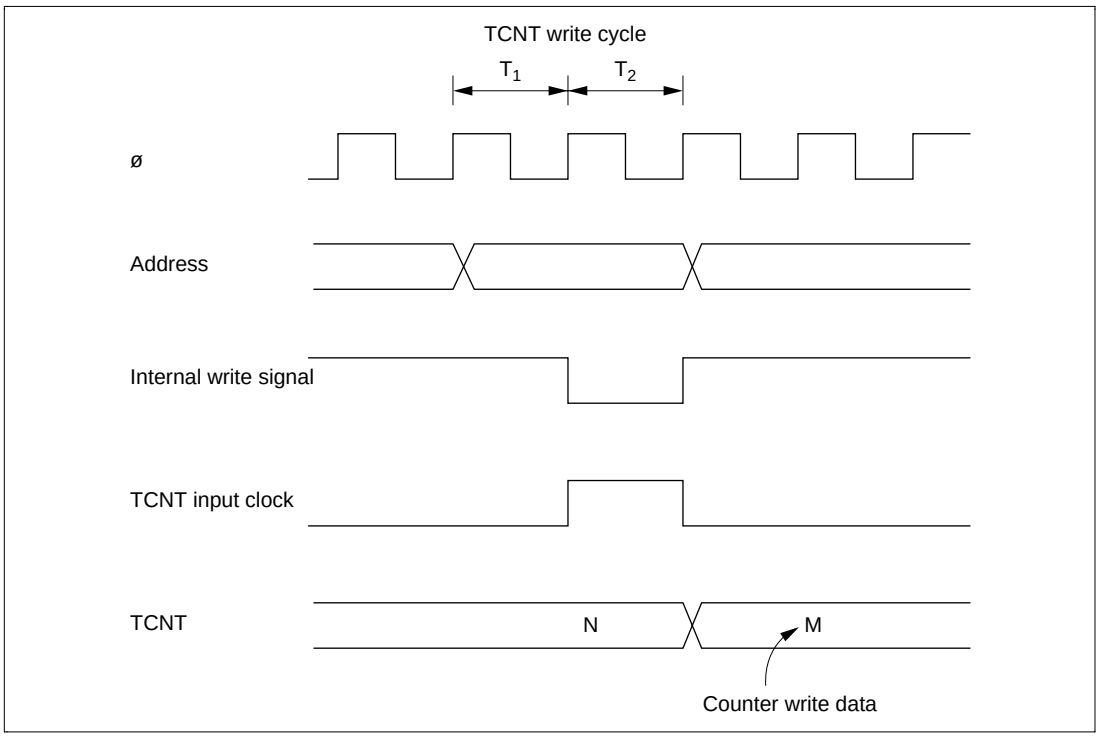



**Figure 13-8 Contention between TCNT Write and Increment**

### 13.5.2 Changing Value of CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the value of bits CKS2 to CKS0.

### 13.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, or vice versa, while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

### 13.5.4 System Reset by $\overline{\text{WDTOVF}}$ Signal

If the $\overline{\text{WDTOVF}}$ output signal is input to the $\overline{\text{RES}}$ pin of the H8S/2655 Series, the H8S/2655 Series will not be initialized correctly. Make sure that the $\overline{\text{WDTOVF}}$ signal is not input logically to the $\overline{\text{RES}}$ pin. To reset the entire system by means of the $\overline{\text{WDTOVF}}$ signal, use the circuit shown in figure 13-9.

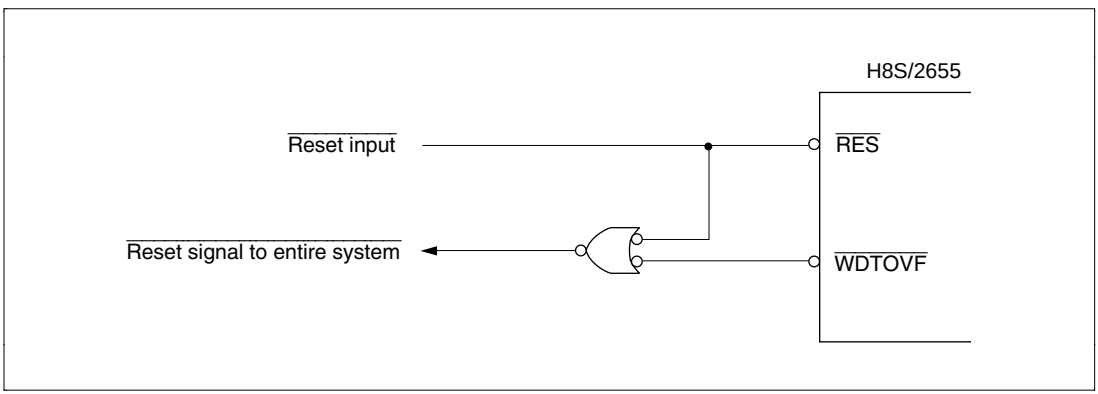



**Figure 13-9 Circuit for System Reset by $\overline{\text{WDTOVF}}$ Signal (Example)**

### 13.5.5 Internal Reset in Watchdog Timer Mode

The H8S/2655 Series is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer operation, but TCNT and TSCR of the WDT are reset.

TCNT, TCSR, and RSTCR cannot be written to while the $\overline{\text{WDTOVF}}$ signal is low. Also note that a read of the WOVF flag is not recognized during this period. To clear the WOVF falg, therefore, read TCSR after the $\overline{\text{WDTOVF}}$ signal goes high, then write 0 to the WOVF flag.

# Section 14 Serial Communication Interface (SCI)

## 14.1 Overview

The H8S/2655 Series is equipped with a 3-channel serial communication interface (SCI). All three channels have the same functions. The SCI can handle both asynchronous and clocked synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

### 14.1.1 Features

SCI features are listed below.

- Choice of asynchronous or clocked synchronous serial communication mode

  Asynchronous mode
  - Serial data communication executed using asynchronous system in which synchronization is achieved character by character

    Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA)
  - A multiprocessor communication function is provided that enables serial data communication with a number of processors
  - Choice of 12 serial data transfer formats

    Data length: 7 or 8 bits

    Stop bit length: 1 or 2 bits

    Parity: Even, odd, or none

    Multiprocessor bit: 1 or 0
  - Receive error detection: Parity, overrun, and framing errors
  - Break detection: Break can be detected by reading the RxD pin level directly in case of a framing error

  Clocked Synchronous mode
  - Serial data communication synchronized with a clock

    Serial data communication can be carried out with other chips that have a synchronous communication function
  - One serial data transfer format

    Data length: 8 bits
  - Receive error detection: Overrun errors detected

- Full-duplex communication capability
  - The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously
  - Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data
- On-chip baud rate generator allows any bit rate to be selected
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin
- Four interrupt sources
  - Four interrupt sources — transmit-data-empty, transmit-end, receive-data-full, and receive error — that can issue requests independently
  - The transmit-data-empty interrupt and receive data full interrupts can activate the DMA controller (DMAC) or data transfer controller (DTC) to execute data transfer
- Choice of LSB-first or MSB-first transfer
  - Can be selected regardless of the communication mode* (except in the case of asynchronous mode bit data)

Note: * Descriptions in this section refer to LSB-first transfer.

### 14.1.2 Block Diagram

Figure 14-1 shows a block diagram of the SCI.

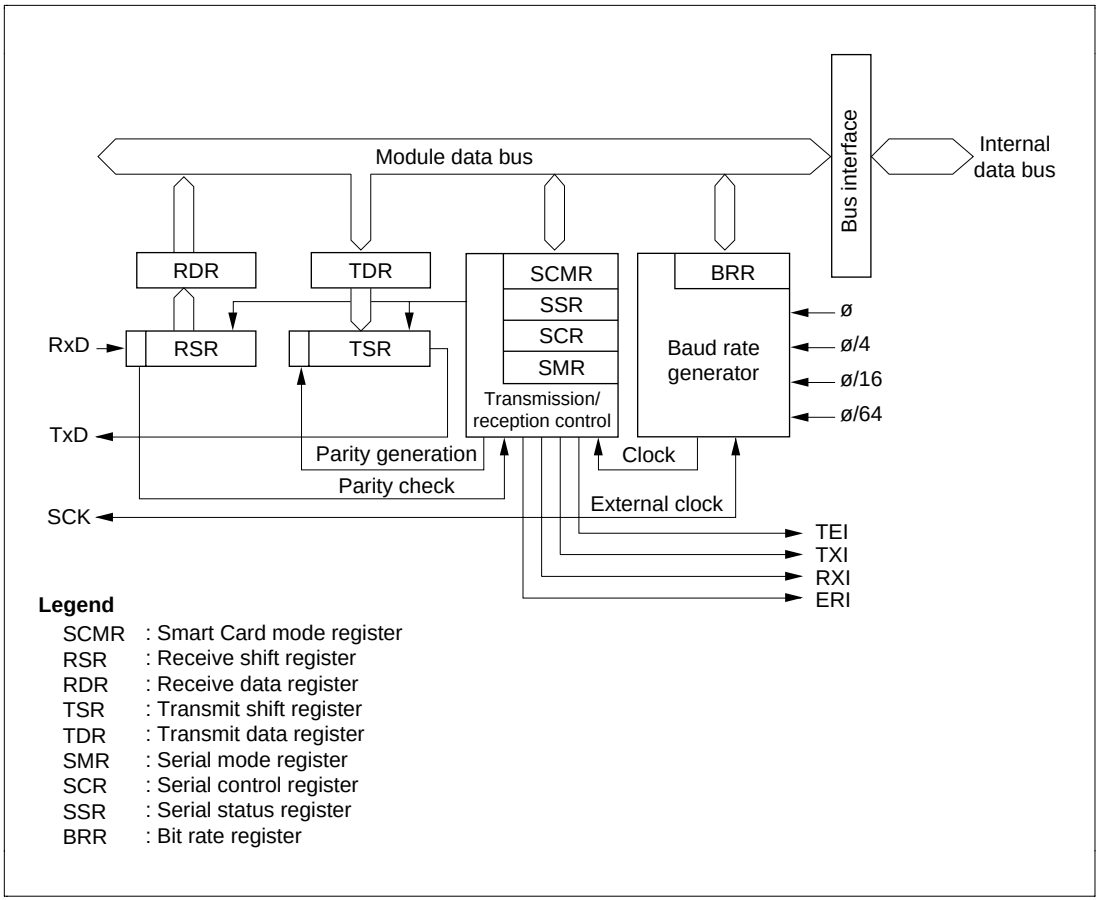



**Figure 14-1 Block Diagram of SCI**

### 14.1.3 Pin Configuration

Table 14-1 shows the serial pins for each SCI channel.

**Table 14-1 SCI Pins**

| Channel | Pin Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Serial clock pin 0 | $SCK_0$ | I/O | SCI0 clock input/output |
| | Receive data pin 0 | $RxD_0$ | Input | SCI0 receive data input |
| | Transmit data pin 0 | $TxD_0$ | Output | SCI0 transmit data output |
| 1 | Serial clock pin 1 | $SCK_1$ | I/O | SCI1 clock input/output |
| | Receive data pin 1 | $RxD_1$ | Input | SCI1 receive data input |
| | Transmit data pin 1 | $TxD_1$ | Output | SCI1 transmit data output |
| 2 | Serial clock pin 2 | $SCK_2$ | I/O | SCI2 clock input/output |
| | Receive data pin 2 | $RxD_2$ | Input | SCI2 receive data input |
| | Transmit data pin 2 | $TxD_2$ | Output | SCI2 transmit data output |

### 14.1.4 Register Configuration

The SCI has the internal registers shown in table 14-2. These registers are used to specify asynchronous mode or clocked synchronous mode, the data format , and the bit rate, and to control transmitter/receiver.

**Table 14-2 SCI Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Serial mode register 0 | SMR0 | R/W | H'00 | H'FF78 |
| | Bit rate register 0 | BRR0 | R/W | H'FF | H'FF79 |
| | Serial control register 0 | SCR0 | R/W | H'00 | H'FF7A |
| | Transmit data register 0 | TDR0 | R/W | H'FF | H'FF7B |
| | Serial status register 0 | SSR0 | R/(W)*[2] | H'84 | H'FF7C |
| | Receive data register 0 | RDR0 | R | H'00 | H'FF7D |
| | Smart card mode register 0 | SCMR0 | R/W | H'F2 | H'FF7E |
| 1 | Serial mode register 1 | SMR1 | R/W | H'00 | H'FF80 |
| | Bit rate register 1 | BRR1 | R/W | H'FF | H'FF81 |
| | Serial control register 1 | SCR1 | R/W | H'00 | H'FF82 |
| | Transmit data register 1 | TDR1 | R/W | H'FF | H'FF83 |
| | Serial status register 1 | SSR1 | R/(W)*[2] | H'84 | H'FF84 |
| | Receive data register 1 | RDR1 | R | H'00 | H'FF85 |
| | Smart card mode register 1 | SCMR1 | R/W | H'F2 | H'FF86 |
| 2 | Serial mode register 2 | SMR2 | R/W | H'00 | H'FF88 |
| | Bit rate register 2 | BRR2 | R/W | H'FF | H'FF89 |
| | Serial control register 2 | SCR2 | R/W | H'00 | H'FF8A |
| | Transmit data register 2 | TDR2 | R/W | H'FF | H'FF8B |
| | Serial status register 2 | SSR2 | R/(W)*[2] | H'84 | H'FF8C |
| | Receive data register 2 | RDR2 | R | H'00 | H'FF8D |
| | Smart card mode register 2 | SCMR2 | R/W | H'F2 | H'FF8E |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 14.2 Register Descriptions

### 14.2.1 Receive Shift Register (RSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| R/W | : | — | — | — | — | — | — | — | — |

RSR is a register used to receive serial data.

The SCI sets serial data input from the RxD pin in RSR in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to RDR automatically.

RSR cannot be directly read or written to by the CPU.

### 14.2.2 Receive Data Register (RDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R | R | R | R | R | R | R | R |

RDR is a register that stores received serial data.

When the SCI has received one byte of serial data, it transfers the received serial data from RSR to RDR where it is stored, and completes the receive operation. After this, RSR is receive-enabled.

Since RSR and RDR function as a double buffer in this way, enables continuous receive operations to be performed.

RDR is a read-only register, and cannot be written to by the CPU.

RDR is initialized to H'00 by a reset, and in standby mode or module stop mode.

### 14.2.3 Transmit Shift Register (TSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| R/W | : | — | — | — | — | — | — | — | — |

TSR is a register used to transmit serial data.

To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from TDR to TSR, and transmission started, automatically. However, data transfer from TDR to TSR is not performed if the TDRE bit in SSR is set to 1.

TSR cannot be directly read or written to by the CPU.

### 14.2.4 Transmit Data Register (TDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TDR is an 8-bit register that stores data for serial transmission.

When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to TDR during serial transmission of the data in TSR.

TDR can be read or written to by the CPU at all times.

TDR is initialized to H'FF by a reset, and in standby mode or module stop mode.

### 14.2.5 Serial Mode Register (SMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SMR is an 8-bit register used to set the SCI's serial transfer format and select the baud rate generator clock source.

SMR can be read or written to by the CPU at all times.

SMR is initialized to H'00 by a reset, and in standby mode or module stop mode.

**Bit 7—Communication Mode (C/$\overline{A}$):** Selects asynchronous mode or clocked synchronous mode as the SCI operating mode.

**Bit 7**

| C/$\overline{A}$ | Description | |
|---|---|---|
| 0 | Asynchronous mode | (Initial value) |
| 1 | Clocked synchronous mode | |

**Bit 6—Character Length (CHR):** Selects 7 or 8 bits as the data length in asynchronous mode. In clocked synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting.

**Bit 6**

| CHR | Description | |
|---|---|---|
| 0 | 8-bit data | (Initial value) |
| 1 | 7-bit data* | |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted, and it is not possible to choose between LSB-first or MSB-first transfer.

**Bit 5—Parity Enable (PE):** In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In clocked synchronous mode, parity bit addition and checking is not performed, regardless of the PE bit setting.

**Bit 5**

| PE | Description | |
|---|---|---|
| 0 | Parity bit addition and checking disabled | (Initial value) |
| 1 | Parity bit addition and checking enabled* | |

Note:* When the PE bit is set to 1, the parity (even or odd) specified by the O/$\overline{E}$ bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/$\overline{E}$ bit.

**Bit 4—Parity Mode (O/$\overline{E}$):** Selects either even or odd parity for use in parity addition and checking.

The O/$\overline{E}$ bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The O/$\overline{E}$ bit setting is invalid in clocked synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

**Bit 4**

| O/$\overline{E}$ | Description | |
|---|---|---|
| 0 | Even parity*[1] | (Initial value) |
| 1 | Odd parity*[2] | |

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is even.
In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is even.
2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is odd.
In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is odd.

**Bit 3—Stop Bit Length (STOP):** Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bits setting is only valid in asynchronous mode. If clocked synchronous mode is set the STOP bit setting is invalid since stop bits are not added.

**Bit 3**

| STOP | Description | |
|---|---|---|
| 0 | 1 stop bit*[1] | (Initial value) |
| 1 | 2 stop bits*[2] | |

Notes: 1. In transmission, a single 1 bit (stop bit) is added to the end of a transmit character before it is sent.
2. In transmission, two 1 bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

**Bit 2—Multiprocessor Mode (MP):** Selects multiprocessor format. When multiprocessor format is selected, the PE bit and O/$\overline{\text{E}}$ bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in clocked synchronous mode.

For details of the multiprocessor communication function, see section 14.3.3, Multiprocessor Communication Function.

**Bit 2**

| MP | Description | |
|---|---|---|
| 0 | Multiprocessor function disabled | (Initial value) |
| 1 | Multiprocessor format selected | |

**Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0):** These bits select the clock source for the baud rate generator. The clock source can be selected from ø, ø/4, ø/16, and ø/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 14.2.8, Bit Rate Register.

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **CKS1** | **CKS0** | **Description** | |
| 0 | 0 | ø clock | (Initial value) |
| | 1 | ø/4 clock | |
| 1 | 0 | ø/16 clock | |
| | 1 | ø/64 clock | |

### 14.2.6 Serial Control Register (SCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SCR is a register that performs enabling or disabling of SCI transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCR can be read or written to by the CPU at all times.

SCR is initialized to H'00 by a reset, and in standby mode or module stop mode.

**Bit 7—Transmit Interrupt Enable (TIE):** Enables or disables transmit data empty interrupt (TXI) request generation when serial transmit data is transferred from TDR to TSR and the TDRE flag in SSR is set to 1.

| Bit 7 | | |
|---|---|---|
| **TIE** | **Description** | |
| 0 | Transmit data empty interrupt (TXI) requests disabled* | (Initial value) |
| 1 | Transmit data empty interrupt (TXI) requests enabled | |

Note:* TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0.

**Bit 6—Receive Interrupt Enable (RIE):** Enables or disables receive data full interrupt (RXI) request and receive error interrupt (ERI) request generation when serial receive data is transferred from RSR to RDR and the RDRF flag in SSR is set to 1.

**Bit 6**

| RIE | Description |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled* (Initial value) |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Note:* RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or clearing the RIE bit to 0.

**Bit 5—Transmit Enable (TE):** Enables or disables the start of serial transmission by the SCI.

**Bit 5**

| TE | Description |
|---|---|
| 0 | Transmission disabled*[1] (Initial value) |
| 1 | Transmission enabled*[2] |

Notes: 1. The TDRE flag in SSR is fixed at 1.
2. In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0.
SMR setting must be performed to decide the transfer format before setting the TE bit to 1.

**Bit 4—Receive Enable (RE):** Enables or disables the start of serial reception by the SCI.

**Bit 4**

| RE | Description |
|---|---|
| 0 | Reception disabled*[1] (Initial value) |
| 1 | Reception enabled*[2] |

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.
2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in clocked synchronous mode.
SMR setting must be performed to decide the transfer format before setting the RE bit to 1.

**Bit 3—Multiprocessor Interrupt Enable (MPIE):** Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when the MP bit in SMR is set to 1.

The MPIE bit setting is invalid in clocked synchronous mode or when the MP bit is cleared to 0.

**Bit 3**

| MPIE | Description | |
|---|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received | (Initial value) |
| 1 | Multiprocessor interrupts enabled*<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received. | |

Note: * When receive data including MPB = 0 is received, receive data transfer from RSR to RDR, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR , is not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR are set to 1) and FER and ORER flag setting is enabled.

**Bit 2—Transmit End Interrupt Enable (TEIE):** Enables or disables transmit end interrupt (TEI) request generation when there is no valid transmit data in TDR in MSB data transmission.

**Bit 2**

| TEIE | Description | |
|---|---|---|
| 0 | Transmit end interrupt (TEI) request disabled* | (Initial value) |
| 1 | Transmit end interrupt (TEI) request enabled* | |

Note: * TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0.

**Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0):** These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin. The combination of the CKE1 and CKE0 bits determines whether the SCK pin functions as an I/O port, the serial clock output pin, or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in clocked synchronous mode, and in the case of external clock operation (CKE1 = 1). Note that the SCI's operating mode must be decided using SMR before setting the CKE1 and CKE0 bits.

For details of clock source selection, see table 14.9 in section 14.3, Operation.

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **CKE1** | **CKE0** | **Description** | |
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port*[1] |
| | | Clocked synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*[2] |
| | | Clocked synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*[3] |
| | | Clocked synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*[3] |
| | | Clocked synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Initial value
2. Outputs a clock of the same frequency as the bit rate.
3. Inputs a clock with a frequency 16 times the bit rate.

### 14.2.7 Serial Status Register (SSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Note: Only 0 can be written, to clear the flag.

SSR is an 8-bit register containing status flags that indicate the operating status of the SCI, and multiprocessor bits.

SSR can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified.

SSR is initialized to H'84 by a reset, and in standby mode or module stop mode.

**Bit 7—Transmit Data Register Empty (TDRE):** Indicates that data has been transferred from TDR to TSR and the next serial data can be written to TDR.

| Bit 7<br>TDRE | Description |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions] (Initial value)<br>• When the TE bit in SCR is 0<br>• When data in transferred from TDR to TSR and data can be written to TDR |

**Bit 6—Receive Data Register Full (RDRF):** Indicates that the received data is stored in RDR.

| Bit 6<br>RDRF | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Note: RDR and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCR is cleared to 0.

If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

**Bit 5—Overrun Error (ORER):** Indicates that an overrun error occurred during reception, causing abnormal termination.

**Bit 5**

| ORER | Description | |
|---|---|---|
| 0 | [Clearing condition] | (Initial value)*[1] |
| | When 0 is written to ORER after reading ORER = 1 | |
| 1 | [Setting condition] | |
| | When the next serial reception is completed while RDRF = 1 | |

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
2. The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 4—Framing Error (FER):** Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

**Bit 4**

| FER | Description | |
|---|---|---|
| 0 | [Clearing condition] | (Initial value)*[1] |
| | When 0 is written to FER after reading FER = 1 | |
| 1 | [Setting condition] | |
| | When the SCI checks whether the stop bit at the end of the receive data when reception ends, and the stop bit is 0 *[2] | |

Notes: 1. The FER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
2. In 2-stop-bit mode, only the first stop bit is checked for a value of 0; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 3—Parity Error (PER):** Indicates that a parity error occurred during reception using parity addition in asynchronous mode, causing abnormal termination.

**Bit 3**

| PER | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*[1]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR*[2] |

Notes: 1. The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.

2. If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 2—Transmit End (TEND):** Indicates that there is no valid data in TDR when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

**Bit 2**

| TEND | Description |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions] (Initial value)<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

**Bit 1—Multiprocessor Bit (MPB):** When reception is performed using multiprocessor format in asynchronous mode, MPB stores the multiprocessor bit in the receive data.

MPB is a read-only bit, and cannot be modified.

**Bit 1**

| MPB | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Note: * Retains its previous state when the RE bit in SCR is cleared to 0 with multiprocessor format.

**Bit 0—Multiprocessor Bit Transfer (MPBT):** When transmission is performed using multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.

The MPBT bit setting is invalid when multiprocessor format is not used, when not transmitting, and in clocked synchronous mode.

**Bit 0**

| MPBT | Description | |
|---|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted | (Initial value) |
| 1 | Data with a 1 multiprocessor bit is transmitted | |

### 14.2.8 Bit Rate Register (BRR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BRR is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SMR.

BRR can be read or written to by the CPU at all times.

BRR is initialized to H'FF by a reset, and in standby mode or module stop mode.

As baud rate generator control is performed independently for each channel, different values can be set for each channel.

Table 14-3 shows sample BRR settings in asynchronous mode, and table 14-4 shows sample BRR settings in clocked synchronous mode.

**Table 14-3 BRR Settings for Various Bit Rates (Asynchronous Mode)**

| | ø = 2 MHz | | | ø = 2.097152 MHz | | | ø = 2.4576 MHz | | | ø = 3 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 1 | 141 | 0.03 | 1 | 148 | –0.04 | 1 | 174 | –0.26 | 1 | 212 | 0.03 |
| 150 | 1 | 103 | 0.16 | 1 | 108 | 0.21 | 1 | 127 | 0.00 | 1 | 155 | 0.16 |
| 300 | 0 | 207 | 0.16 | 0 | 217 | 0.21 | 0 | 255 | 0.00 | 1 | 77 | 0.16 |
| 600 | 0 | 103 | 0.16 | 0 | 108 | 0.21 | 0 | 127 | 0.00 | 0 | 155 | 0.16 |
| 1200 | 0 | 51 | 0.16 | 0 | 54 | –0.70 | 0 | 63 | 0.00 | 0 | 77 | 0.16 |
| 2400 | 0 | 25 | 0.16 | 0 | 26 | 1.14 | 0 | 31 | 0.00 | 0 | 38 | 0.16 |
| 4800 | 0 | 12 | 0.16 | 0 | 13 | –2.48 | 0 | 15 | 0.00 | 0 | 19 | –2.34 |
| 9600 | 0 | 6 | –6.99 | 0 | 6 | –2.48 | 0 | 7 | 0.00 | 0 | 9 | –2.34 |
| 19200 | 0 | 2 | 8.51 | 0 | 2 | 13.78 | 0 | 3 | 0.00 | 0 | 4 | –2.34 |
| 31250 | 0 | 1 | 0.00 | 0 | 1 | 4.86 | 0 | 1 | 22.88 | 0 | 2 | 0.00 |
| 38400 | 0 | 1 | –18.62 | 0 | 1 | –14.67 | 0 | 1 | 0.00 | — | — | — |

| | ø = 3.6864 MHz | | | ø = 4 MHz | | | ø = 4.9152 MHz | | | ø = 5 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 64 | 0.70 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 88 | –0.25 |
| 150 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 300 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 600 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 1200 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 2400 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 4800 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 9600 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 19200 | 0 | 5 | 0.00 | 0 | 6 | –6.99 | 0 | 7 | 0.00 | 0 | 7 | 1.73 |
| 31250 | — | — | — | 0 | 3 | 0.00 | 0 | 4 | –1.70 | 0 | 4 | 0.00 |
| 38400 | 0 | 2 | 0.00 | 0 | 2 | 8.51 | 0 | 3 | 0.00 | 0 | 3 | 1.73 |

**Table 14-3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

| | ø = 6 MHz | | | ø = 6.144 MHz | | | ø = 7.3728 MHz | | | ø = 8 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 106 | –0.44 | 2 | 108 | 0.08 | 2 | 130 | –0.07 | 2 | 141 | 0.03 |
| 150 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 95 | 0.00 | 2 | 103 | 0.16 |
| 300 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 191 | 0.00 | 1 | 207 | 0.16 |
| 600 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 95 | 0.00 | 1 | 103 | 0.16 |
| 1200 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 191 | 0.00 | 0 | 207 | 0.16 |
| 2400 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 95 | 0.00 | 0 | 103 | 0.16 |
| 4800 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 47 | 0.00 | 0 | 51 | 0.16 |
| 9600 | 0 | 19 | –2.34 | 0 | 19 | 0.00 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 19200 | 0 | 9 | –2.34 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 12 | 0.16 |
| 31250 | 0 | 5 | 0.00 | 0 | 5 | 2.40 | 0 | 6 | 5.33 | 0 | 7 | 0.00 |
| 38400 | 0 | 4 | –2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 | 0 | 6 | –6.99 |

| | ø = 9.8304 MHz | | | ø = 10 MHz | | | ø = 12 MHz | | | ø = 12.288 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 174 | –0.26 | 2 | 177 | –0.25 | 2 | 212 | 0.03 | 2 | 217 | 0.08 |
| 150 | 2 | 127 | 0.00 | 2 | 129 | 0.16 | 2 | 155 | 0.16 | 2 | 159 | 0.00 |
| 300 | 1 | 255 | 0.00 | 2 | 64 | 0.16 | 2 | 77 | 0.16 | 2 | 79 | 0.00 |
| 600 | 1 | 127 | 0.00 | 1 | 129 | 0.16 | 1 | 155 | 0.16 | 1 | 159 | 0.00 |
| 1200 | 0 | 255 | 0.00 | 1 | 64 | 0.16 | 1 | 77 | 0.16 | 1 | 79 | 0.00 |
| 2400 | 0 | 127 | 0.00 | 0 | 129 | 0.16 | 0 | 155 | 0.16 | 0 | 159 | 0.00 |
| 4800 | 0 | 63 | 0.00 | 0 | 64 | 0.16 | 0 | 77 | 0.16 | 0 | 79 | 0.00 |
| 9600 | 0 | 31 | 0.00 | 0 | 32 | –1.36 | 0 | 38 | 0.16 | 0 | 39 | 0.00 |
| 19200 | 0 | 15 | 0.00 | 0 | 15 | 1.73 | 0 | 19 | –2.34 | 0 | 19 | 0.00 |
| 31250 | 0 | 9 | –1.70 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 11 | 2.40 |
| 38400 | 0 | 7 | 0.00 | 0 | 7 | 1.73 | 0 | 9 | –2.34 | 0 | 9 | 0.00 |

**Table 14-3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

| | ø = 14 MHz | | | ø = 14.7456 MHz | | | ø = 16 MHz | | | ø = 17.2032 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 248 | –0.17 | 3 | 64 | 0.70 | 3 | 70 | 0.03 | 3 | 75 | 0.48 |
| 150 | 2 | 181 | 0.16 | 2 | 191 | 0.00 | 2 | 207 | 0.16 | 2 | 223 | 0.00 |
| 300 | 2 | 90 | 0.16 | 2 | 95 | 0.00 | 2 | 103 | 0.16 | 2 | 111 | 0.00 |
| 600 | 1 | 181 | 0.16 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 223 | 0.00 |
| 1200 | 1 | 90 | 0.16 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 111 | 0.00 |
| 2400 | 0 | 181 | 0.16 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 223 | 0.00 |
| 4800 | 0 | 90 | 0.16 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 111 | 0.00 |
| 9600 | 0 | 45 | –0.93 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 55 | 0.00 |
| 19200 | 0 | 22 | –0.93 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 27 | 0.00 |
| 31250 | 0 | 13 | 0.00 | 0 | 14 | –1.70 | 0 | 15 | 0.00 | 0 | 16 | 1.20 |
| 38400 | 0 | 10 | 3.57 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 13 | 0.00 |

| | ø = 18 MHz | | | ø = 19.6608 MHz | | | ø = 20 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 3 | 79 | –0.12 | 3 | 86 | 0.31 | 3 | 88 | –0.25 |
| 150 | 2 | 233 | 0.16 | 2 | 255 | 0.00 | 3 | 64 | 0.16 |
| 300 | 2 | 116 | 0.16 | 2 | 127 | 0.00 | 2 | 129 | 0.16 |
| 600 | 1 | 233 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 1200 | 1 | 116 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 2400 | 0 | 233 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 4800 | 0 | 116 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 9600 | 0 | 58 | –0.69 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 19200 | 0 | 28 | 1.02 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 31250 | 0 | 17 | 0.00 | 0 | 19 | –1.70 | 0 | 19 | 0.00 |
| 38400 | 0 | 14 | –2.34 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |

**Table 14-4 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)**

| Bit Rate | ø = 2 MHz | | ø = 4 MHz | | ø = 8 MHz | | ø = 10 MHz | | ø = 16 MHz | | ø = 20 MHz | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (bit/s) | n | N | n | N | n | N | n | N | n | N | n | N |
| 110 | 3 | 70 | — | — | — | — | — | — | — | — | — | — |
| 250 | 2 | 124 | 2 | 249 | 3 | 124 | — | — | 3 | 249 | — | — |
| 500 | 1 | 249 | 2 | 124 | 2 | 249 | — | — | 3 | 124 | — | — |
| 1 k | 1 | 124 | 1 | 249 | 2 | 124 | — | — | 2 | 249 | — | — |
| 2.5 k | 0 | 199 | 1 | 99 | 1 | 199 | 1 | 249 | 2 | 99 | 2 | 124 |
| 5 k | 0 | 99 | 0 | 199 | 1 | 99 | 1 | 124 | 1 | 199 | 1 | 249 |
| 10 k | 0 | 49 | 0 | 99 | 0 | 199 | 0 | 249 | 1 | 99 | 1 | 124 |
| 25 k | 0 | 19 | 0 | 39 | 0 | 79 | 0 | 99 | 0 | 159 | 0 | 199 |
| 50 k | 0 | 9 | 0 | 19 | 0 | 39 | 0 | 49 | 0 | 79 | 0 | 99 |
| 100 k | 0 | 4 | 0 | 9 | 0 | 19 | 0 | 24 | 0 | 39 | 0 | 49 |
| 250 k | 0 | 1 | 0 | 3 | 0 | 7 | 0 | 9 | 0 | 15 | 0 | 19 |
| 500 k | 0 | 0* | 0 | 1 | 0 | 3 | 0 | 4 | 0 | 7 | 0 | 9 |
| 1 M | | | 0 | 0* | 0 | 1 | — | — | 0 | 3 | 0 | 4 |
| 2.5 M | | | | | — | — | 0 | 0* | — | — | 0 | 1 |
| 5 M | | | | | | | | | — | — | 0 | 0* |

Note: As far as possible, the setting should be made so that the error is no more than 1%.

**Legend**

Blank : Cannot be set.

— : Can be set, but there will be a degree of error.

* : Continuous transfer is not possible.

The BRR setting is found from the following formulas.

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Clocked synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bit/s)
N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
ø: Operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

| | | SMR Setting | |
|---|---|---|---|
| **n** | **Clock** | **CKS1** | **CKS0** |
| 0 | ø | 0 | 0 |
| 1 | ø/4 | 0 | 1 |
| 2 | ø/16 | 1 | 0 |
| 3 | ø/64 | 1 | 1 |

The bit rate error in asynchronous mode is found from the following formula:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 14-5 shows the maximum bit rate for each frequency in asynchronous mode. Tables 14-6 and 14-7 show the maximum bit rates with external clock input.

**Table 14-5 Maximum Bit Rate for Each Frequency (Asynchronous Mode)**

| ø (MHz) | Maximum Bit Rate (bit/s) | n | N |
|---|---|---|---|
| 2 | 62500 | 0 | 0 |
| 2.097152 | 65536 | 0 | 0 |
| 2.4576 | 76800 | 0 | 0 |
| 3 | 93750 | 0 | 0 |
| 3.6864 | 115200 | 0 | 0 |
| 4 | 125000 | 0 | 0 |
| 4.9152 | 153600 | 0 | 0 |
| 5 | 156250 | 0 | 0 |
| 6 | 187500 | 0 | 0 |
| 6.144 | 192000 | 0 | 0 |
| 7.3728 | 230400 | 0 | 0 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 10 | 312500 | 0 | 0 |
| 12 | 375000 | 0 | 0 |
| 12.288 | 384000 | 0 | 0 |
| 14 | 437500 | 0 | 0 |
| 14.7456 | 460800 | 0 | 0 |
| 16 | 500000 | 0 | 0 |
| 17.2032 | 537600 | 0 | 0 |
| 18 | 562500 | 0 | 0 |
| 19.6608 | 614400 | 0 | 0 |
| 20 | 625000 | 0 | 0 |

**Table 14-6 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

| ø (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bit/s) |
|---|---|---|
| 2 | 0.5000 | 31250 |
| 2.097152 | 0.5243 | 32768 |
| 2.4576 | 0.6144 | 38400 |
| 3 | 0.7500 | 46875 |
| 3.6864 | 0.9216 | 57600 |
| 4 | 1.0000 | 62500 |
| 4.9152 | 1.2288 | 76800 |
| 5 | 1.2500 | 78125 |
| 6 | 1.5000 | 93750 |
| 6.144 | 1.5360 | 96000 |
| 7.3728 | 1.8432 | 115200 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 10 | 2.5000 | 156250 |
| 12 | 3.0000 | 187500 |
| 12.288 | 3.0720 | 192000 |
| 14 | 3.5000 | 218750 |
| 14.7456 | 3.6864 | 230400 |
| 16 | 4.0000 | 250000 |
| 17.2032 | 4.3008 | 268800 |
| 18 | 4.5000 | 281250 |
| 19.6608 | 4.9152 | 307200 |
| 20 | 5.0000 | 312500 |

**Table 14-7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)**

| ø (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bit/s) |
|---|---|---|
| 2 | 0.3333 | 333333.3 |
| 4 | 0.6667 | 666666.7 |
| 6 | 1.0000 | 1000000.0 |
| 8 | 1.3333 | 1333333.3 |
| 10 | 1.6667 | 1666666.7 |
| 12 | 2.0000 | 2000000.0 |
| 14 | 2.3333 | 2333333.3 |
| 16 | 2.6667 | 2666666.7 |
| 18 | 3.0000 | 3000000.0 |
| 20 | 3.3333 | 3333333.3 |

### 14.2.9 Smart Card Mode Register (SCMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | — | R/W |

SCMR selects LSB-first or MSB-first by means of bit SDIR. Except in the case of asynchronous mode 7-bit data, LSB-first or MSB-first can be selected regardless of the serial communication mode. The descriptions in this chapter refer to LSB-first transfer.

For details of the other bits in SCMR, see 15.2.1, Smart Card Mode Register (SCMR).

SCMR is initialized to H'F2 by a reset, and in standby mode or module stop mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format. This bit is valid when 8-bit data is used as the transmit/receive format.

**Bit 3**

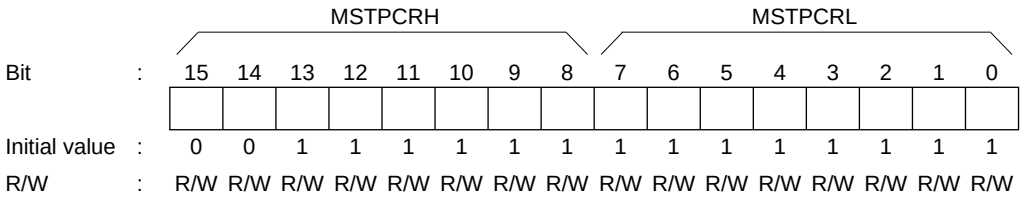
| SDIR | Description | |
|---|---|---|
| 0 | TDR contents are transmitted LSB-first | (Initial value) |
| | Receive data is stored in RDR LSB-first | |
| 1 | TDR contents are transmitted MSB-first | |
| | Receive data is stored in RDR MSB-first | |

**Bit 2—Smart Card Data Invert (SINV):** When the smart card interface operates as a normal SCI, 0 should be written in this bit.

**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** When the smart card interface operates as a normal SCI, 0 should be written in this bit.

### 14.2.10 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the corresponding bit of bits MSTP7 to MSTP5 is set to 1, SCI operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Module Stop (MSTP7):** Specifies the SCI channel 2 module stop mode.

**Bit 7**

| MSTP7 | Description | |
|---|---|---|
| 0 | SCI channel 2 module stop mode cleared | |
| 1 | SCI channel 2 module stop mode set | (Initial value) |

**Bit 6—Module Stop (MSTP6):** Specifies the SCI channel 1 module stop mode.

**Bit 6**

| MSTP6 | Description | |
|---|---|---|
| 0 | SCI channel 1 module stop mode cleared | |
| 1 | SCI channel 1 module stop mode set | (Initial value) |

**Bit 5—Module Stop (MSTP5):** Specifies the SCI channel 0 module stop mode.

**Bit 5**

| MSTP5 | Description | |
|---|---|---|
| 0 | SCI channel 0 module stop mode cleared | |
| 1 | SCI channel 0 module stop mode set | (Initial value) |

## 14.3 Operation

### 14.3.1 Overview

The SCI can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and clocked synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or clocked synchronous mode and the transmission format is made using SMR as shown in table 14-8. The SCI clock is determined by a combination of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR, as shown in table 14-9.

**Asynchronous Mode**

- Data length: Choice of 7 or 8 bits
- Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing, parity, and overrun errors, and breaks, during reception
- Choice of internal or external clock as SCI clock source
  - — When internal clock is selected:

    The SCI operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output
  - — When external clock is selected:

    A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used)

**Clocked Synchronous Mode**

- Transfer format: Fixed 8-bit data
- Detection of overrun errors during reception
- Choice of internal or external clock as SCI clock source
  - — When internal clock is selected:

    The SCI operates on the baud rate generator clock and a serial clock is output off-chip
  - — When external clock is selected:

    The on-chip baud rate generator is not used, and the SCI operates on the input serial clock

**Table 14-8 SMR Settings and Serial Transfer Format Selection**

| SMR Settings | | | | | | SCI Transfer Format | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 2 | Bit 5 | Bit 3 | | | Multi Processor | | |
| C/$\overline{A}$ | CHR | MP | PE | STOP | Mode | Data Length | Bit | Parity Bit | Stop Bit Length |
| 0 | 0 | 0 | 0 | 0 | Asynchronous mode | 8-bit data | No | No | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | | | 1 | 0 | | | | Yes | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | 1 | | 0 | 0 | | 7-bit data | | No | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | | | 1 | 0 | | | | Yes | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | 0 | 1 | — | 0 | Asynchronous mode (multi-processor format) | 8-bit data | Yes | No | 1 bit |
| | | | — | 1 | | | | | 2 bits |
| | 1 | | — | 0 | | 7-bit data | | | 1 bit |
| | | | — | 1 | | | | | 2 bits |
| 1 | — | — | — | — | Clocked synchronous mode | 8-bit data | No | | None |

**Table 14-9 SMR and SCR Settings and SCI Clock Source Selection**

| SMR | SCR Setting | | | SCI Transmit/Receive Clock | |
|---|---|---|---|---|---|
| Bit 7 | Bit 1 | Bit 0 | | Clock | |
| C/$\overline{A}$ | CKE1 | CKE0 | Mode | Source | SCK Pin Function |
| 0 | 0 | 0 | Asynchronous mode | Internal | SCI does not use SCK pin |
| | | 1 | | | Outputs clock with same frequency as bit rate |
| | 1 | 0 | | External | Inputs clock with frequency of 16 times the bit rate |
| | | 1 | | | |
| 1 | 0 | 0 | Clocked synchronous mode | Internal | Outputs serial clock |
| | | 1 | | | |
| | 1 | 0 | | External | Inputs serial clock |
| | | 1 | | | |

### 14.3.2 Operation in Asynchronous Mode

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 14-2 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI performs synchronization at the falling edge of the start bit in reception. The SCI samples the data on the 8th pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.

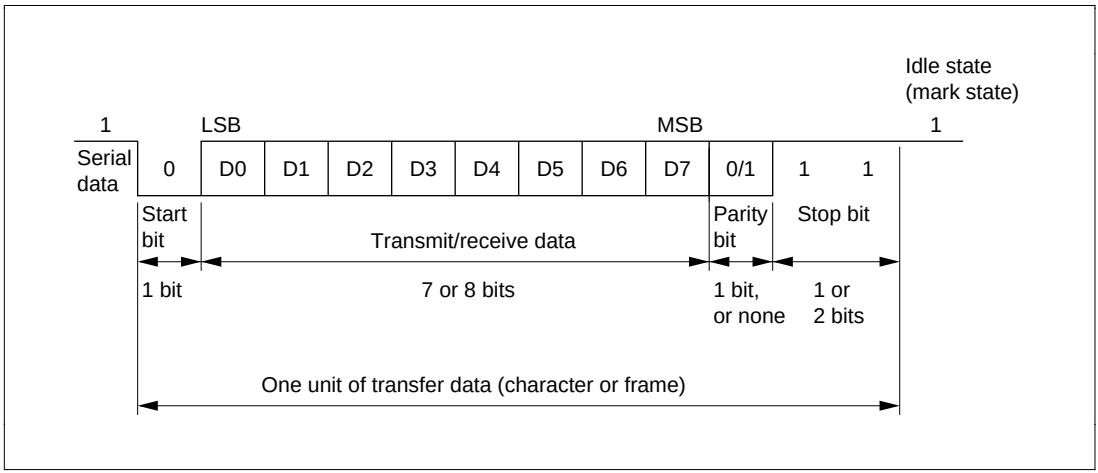



**Figure 14-2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)**

**Data Transfer Format:** Table 14-10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting.

**Table 14-10 Serial Transfer Formats (Asynchronous Mode)**

| SMR Settings | | | | Serial Transfer Format and Frame Length |
|---|---|---|---|---|
| CHR | PE | MP | STOP | 1 2 3 4 5 6 7 8 9 10 11 12 |
| 0 | 0 | 0 | 0 | S 8-bit data STOP |
| 0 | 0 | 0 | 1 | S 8-bit data STOP STOP |
| 0 | 1 | 0 | 0 | S 8-bit data P STOP |
| 0 | 1 | 0 | 1 | S 8-bit data P STOP STOP |
| 1 | 0 | 0 | 0 | S 7-bit data STOP |
| 1 | 0 | 0 | 1 | S 7-bit data STOP STOP |
| 1 | 1 | 0 | 0 | S 7-bit data P STOP |
| 1 | 1 | 0 | 1 | S 7-bit data P STOP STOP |
| 0 | — | 1 | 0 | S 8-bit data MPB STOP |
| 0 | — | 1 | 1 | S 8-bit data MPB STOP STOP |
| 1 | — | 1 | 0 | S 7-bit data MPB STOP |
| 1 | — | 1 | 1 | S 7-bit data MPB STOP STOP |

**Legend**

S : Start bit
STOP : Stop bit
P : Parity bit
MPB : Multiprocessor bit

**Clock:** Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the SCI's serial clock, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 14-9.

When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 14-3.

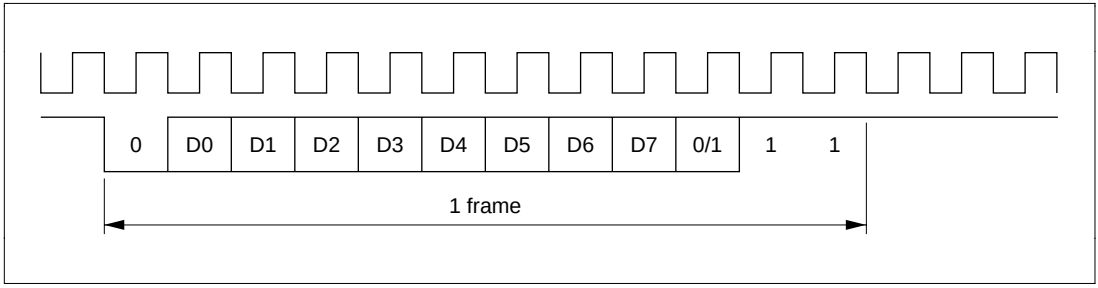



**Figure 14-3 Relation between Output Clock and Transfer Data Phase (Asynchronous Mode)**

**Data Transfer Operations:**

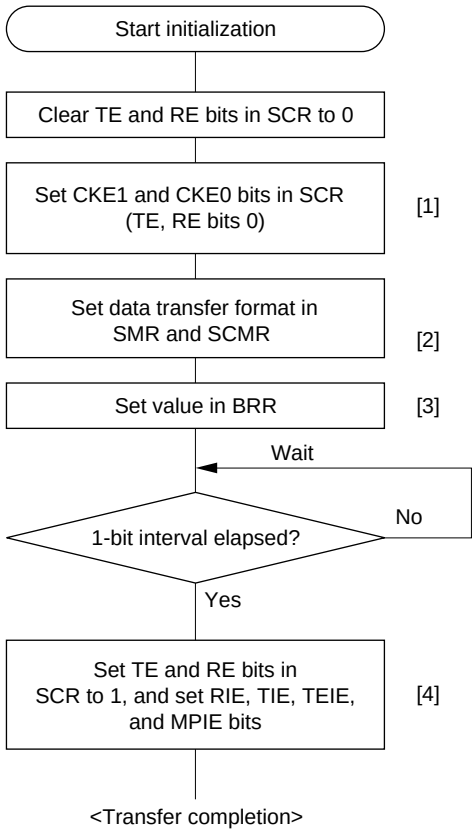
- SCI initialization (asynchronous mode)

  Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

  When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

  When an external clock is used the clock should not be stopped during operation, including initialization, since operation is uncertain.

Figure 14-4 shows a sample SCI initialization flowchart.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock is selected in asynchronous mode, it is output immediately after SCR settings are made.

[2] Set the data transfer format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.

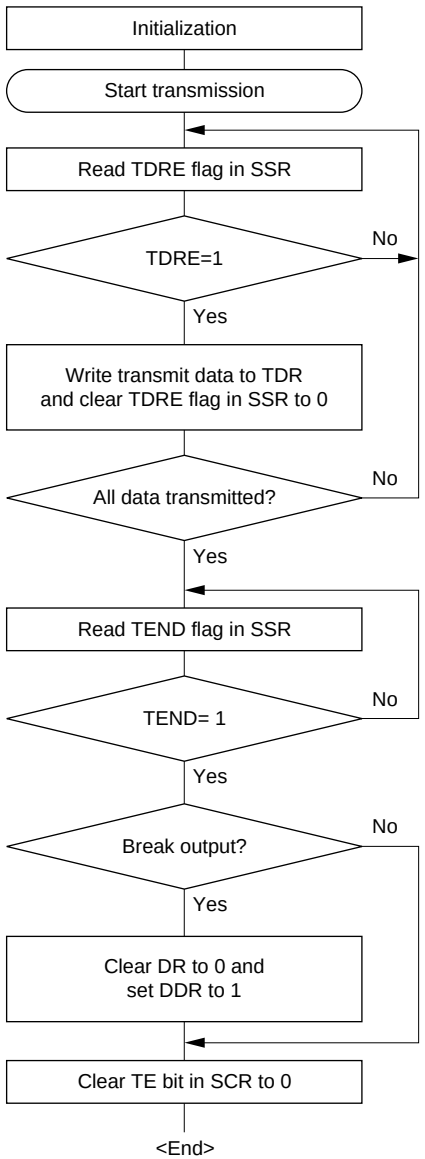
Setting the TE and RE bits enables the TxD and RxD pins to be used.

**Figure 14-4 Sample SCI Initialization Flowchart**

- Serial data transmission (asynchronous mode)

  Figure 14-5 shows a sample flowchart for serial transmission.

  The following procedure should be used for serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit data empty interrupt (TXI) request, and data is written to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 14-5 Sample Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:
One 0-bit is output.

[b] Transmit data:
8-bit or 7-bit data is output in LSB-first order.

[c] Parity bit or multiprocessor bit:
One parity bit (even or odd parity), or one multiprocessor bit is output.
A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.

[d] Stop bit(s):
One or two 1-bits (stop bits) are output.

[e] Mark state:
1 is output continuously until the start bit that starts the next transmission is sent.

[3] The SCI checks the TDRE flag at the timing for sending the stop bit.
If the TDRE flag is cleared to 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the “mark state” is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 14-6 shows an example of the operation for transmission in asynchronous mode.

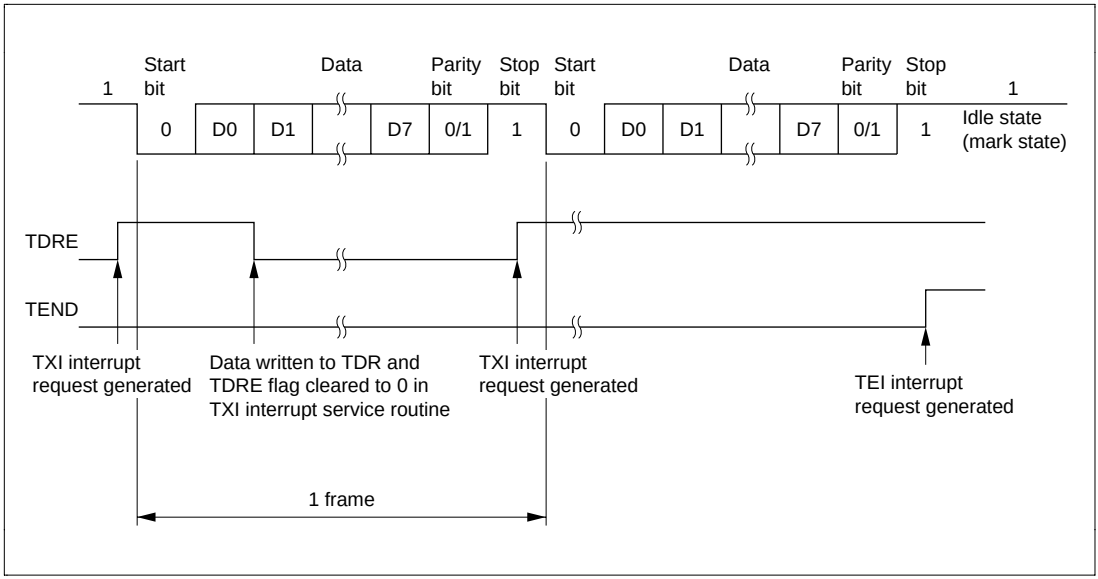
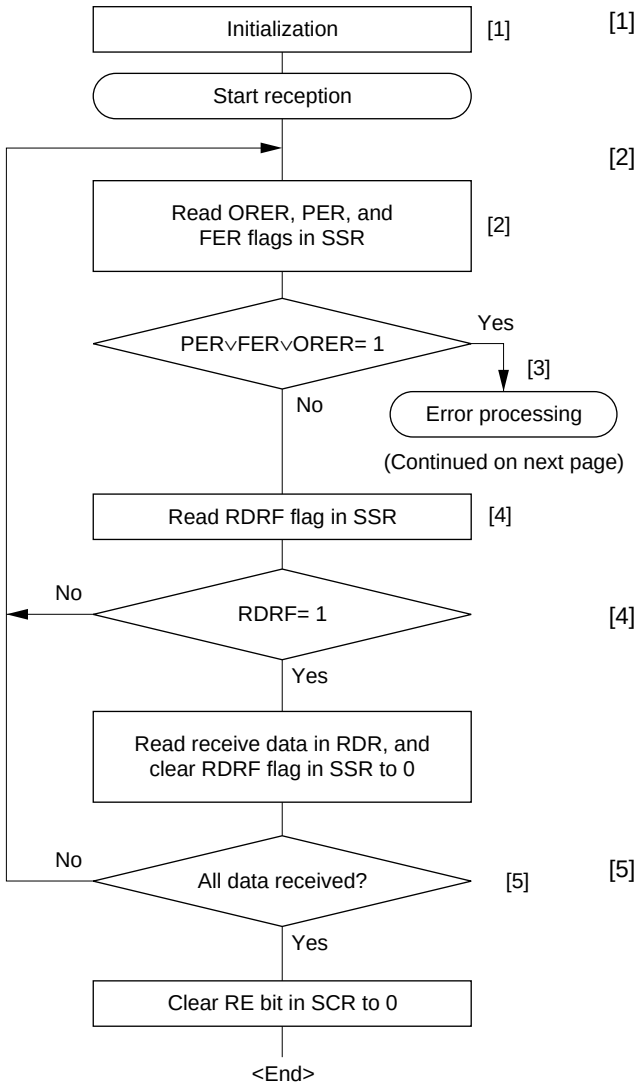



**Figure 14-6 Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)**

- Serial data reception (asynchronous mode)

  Figure 14-7 shows a sample flowchart for serial reception.

  The following procedure should be used for serial data reception.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI status check and receive data read :
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag, read RDR, and clear the RDRF flag to 0. The RDRF flag is cleared automatically when the DMAC or DTC is activated by an RXI interrupt and the RDR value is read.

**Figure 14-7 Sample Serial Reception Data Flowchart**

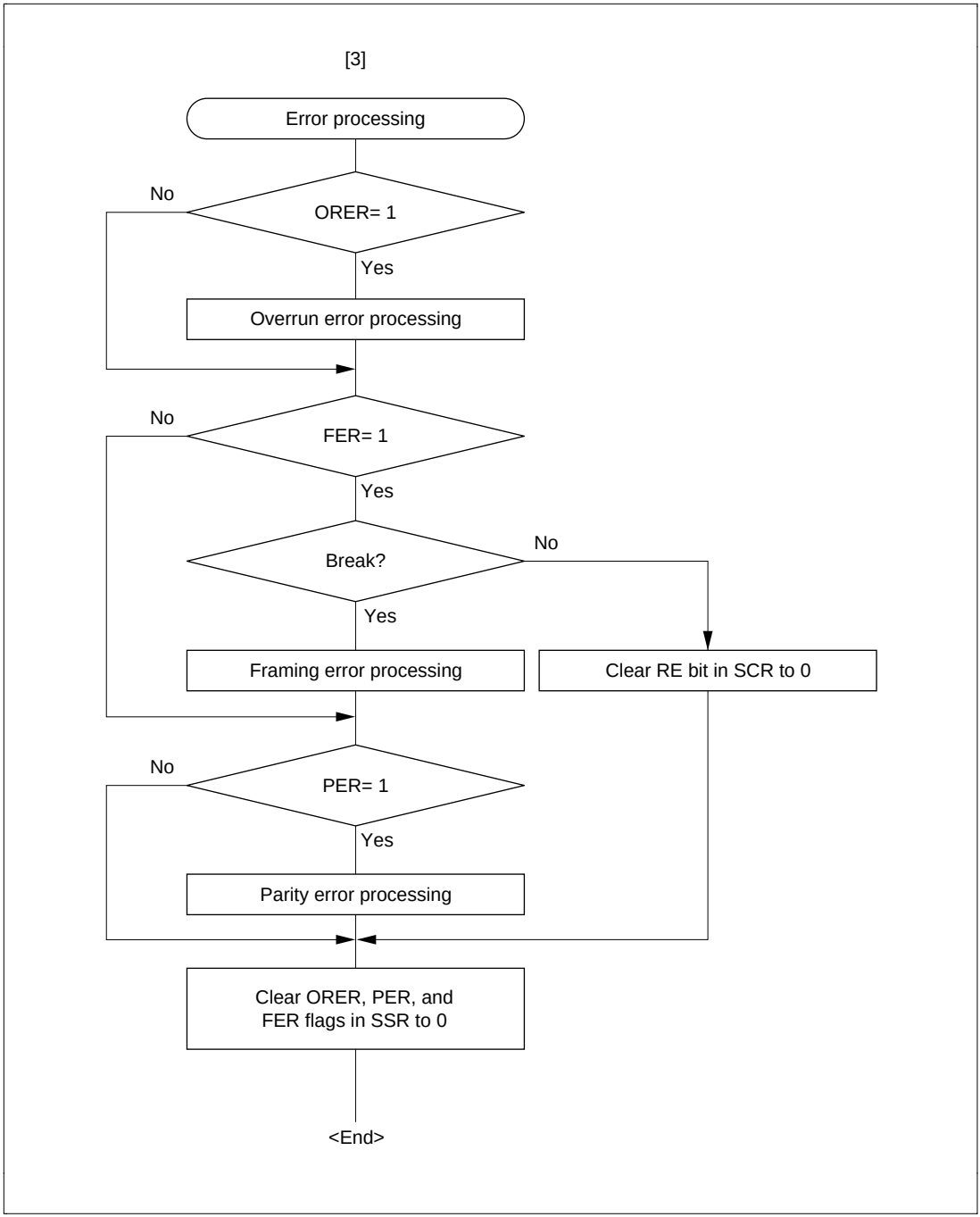



**Figure 14-7 Sample Serial Reception Data Flowchart (cont)**

In serial reception, the SCI operates as described below.

[1] The SCI monitors the transmission line, and if a 0 stop bit is detected, performs internal synchronization and starts reception.

[2] The received data is stored in RSR in LSB-to-MSB order.

[3] The parity bit and stop bit are received.

After receiving these bits, the SCI carries out the following checks.

[a] Parity check:

The SCI checks whether the number of 1 bits in the receive data agrees with the parity (even or odd) set in the O/$\overline{E}$ bit in SMR.

[b] Stop bit check:

The SCI checks whether the stop bit is 1.

If there are two stop bits, only the first is checked.

[c] Status check:

The SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from RSR to RDR.

If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in RDR.

If a receive error* is detected in the error check, the operation is as shown in table 14-11.

Note: * Subsequent receive operations cannot be performed when a receive error has occurred. Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

[4] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive data full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER, PER, or FER flag changes to 1, a receive error interrupt (ERI) request is generated.

**Table 14-11 Receive Errors and Conditions for Occurrence**

| Receive Error | Abbreviation | Occurrence Condition | Data Transfer |
|---|---|---|---|
| Overrun error | ORER | When the next data reception is completed while the RDRF flag in SSR is set to 1 | Receive data is not transferred from RSR to RDR. |
| Framing error | FER | When the stop bit is 0 | Receive data is transferred from RSR to RDR. |
| Parity error | PER | When the received data differs from the parity (even or odd) set in SMR | Receive data is transferred from RSR to RDR. |

Figure 14-8 shows an example of the operation for reception in asynchronous mode.

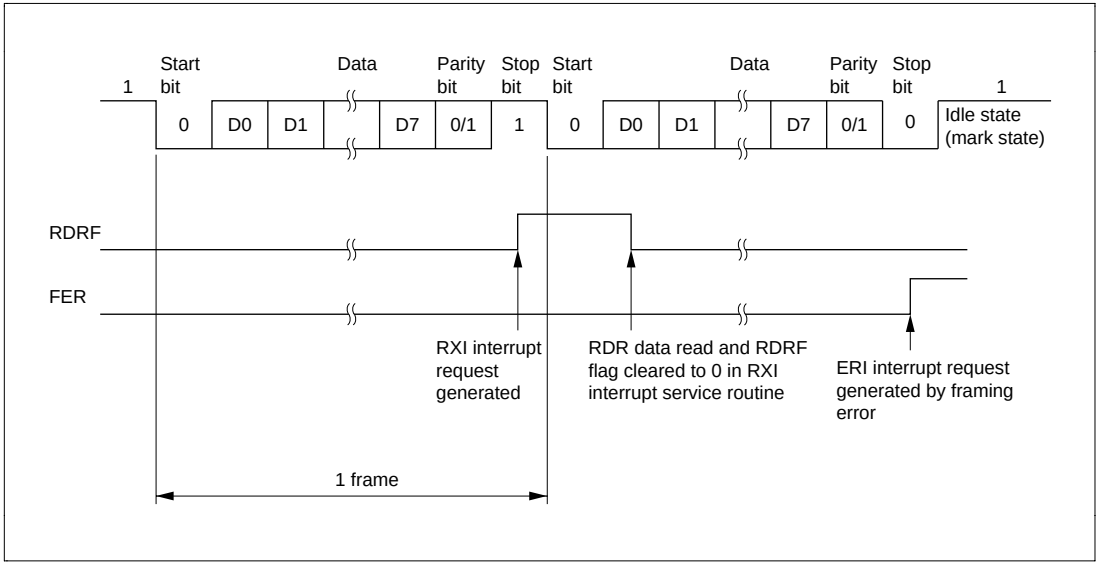



**Figure 14-8 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)**

### 14.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing transmission lines.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station , and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added.

The receiving station skips the data until data with a 1 multiprocessor bit is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

Figure 14-9 shows an example of inter-processor communication using the multiprocessor format.

**Data Transfer Format:** There are four data transfer formats.

When the multiprocessor format is specified, the parity bit specification is invalid.

For details, see table 14-10.

**Clock:** See the section on asynchronous mode.

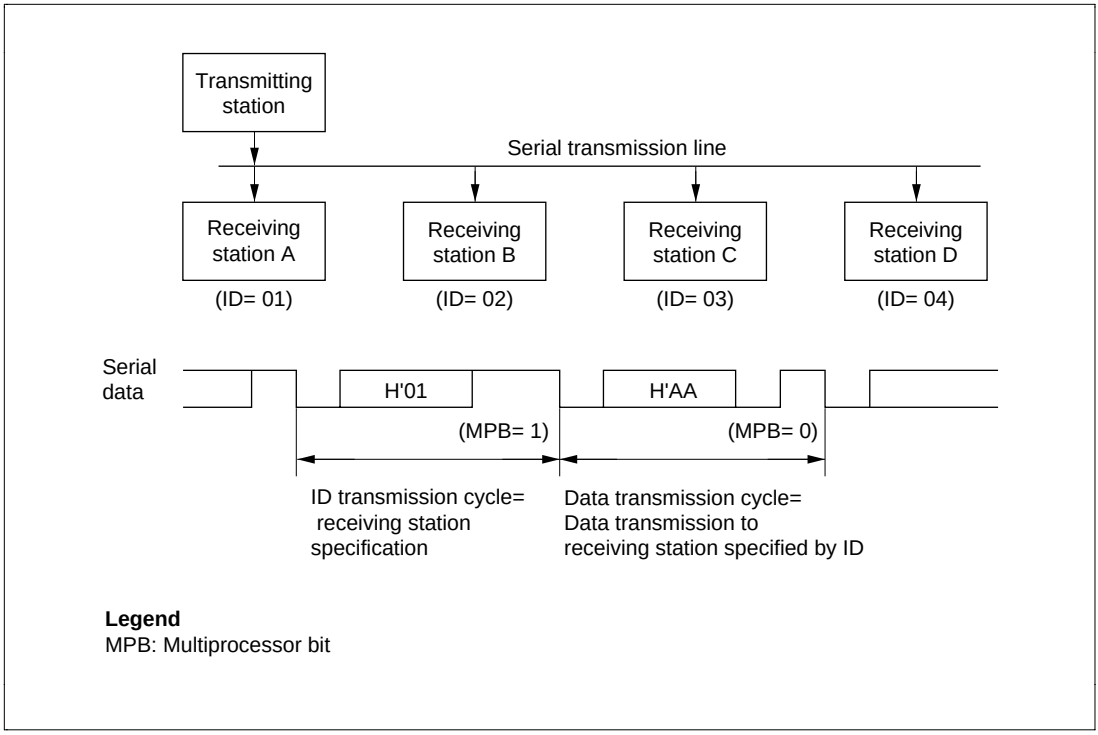


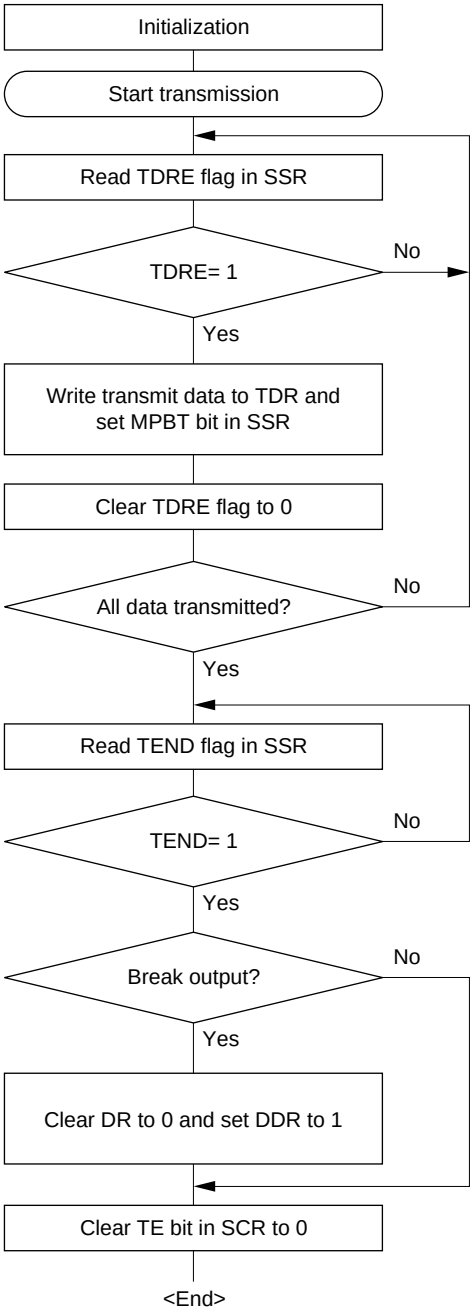

**Figure 14-9 Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)**

**Data Transfer Operations:**

- Multiprocessor serial data transmission

  Figure 14-10 shows a sample flowchart for multiprocessor serial data transmission.

  The following procedure should be used for multiprocessor serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1.
Finally, clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit data empty interrupt (TXI) request, and data is written to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set the port DDR to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 14-10 Sample Multiprocessor Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Multiprocessor bit

One multiprocessor bit (MPBT value) is output.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

1 is output continuously until the start bit that starts the next transmission is sent.

[3] The SCI checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a transmission end interrupt (TEI) request is generated.

Figure 14-11 shows an example of SCI operation for transmission using the multiprocessor format.

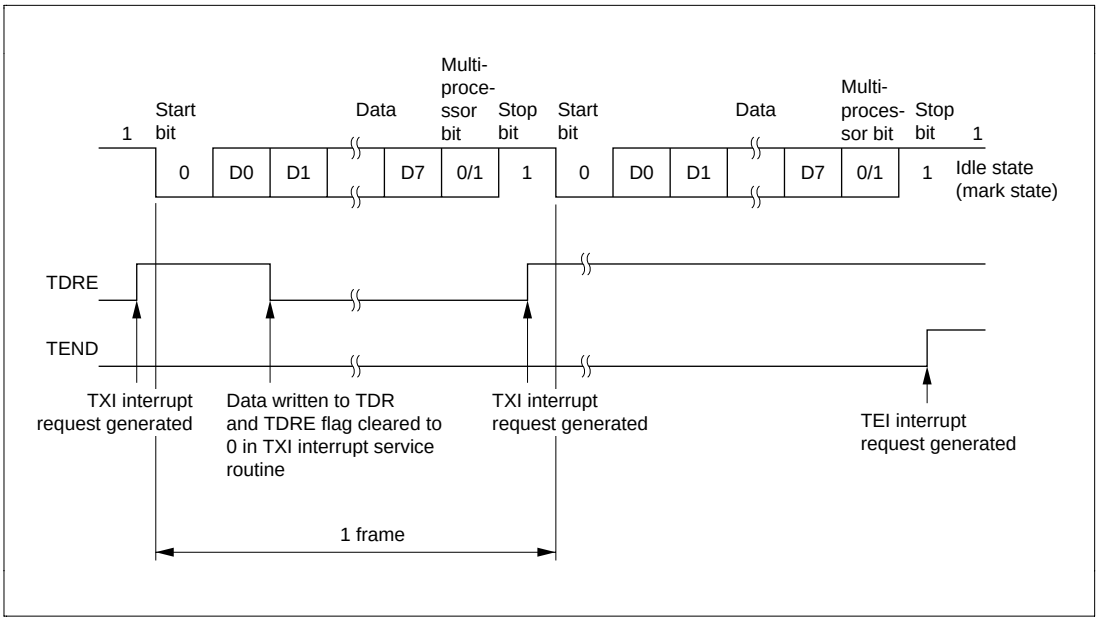
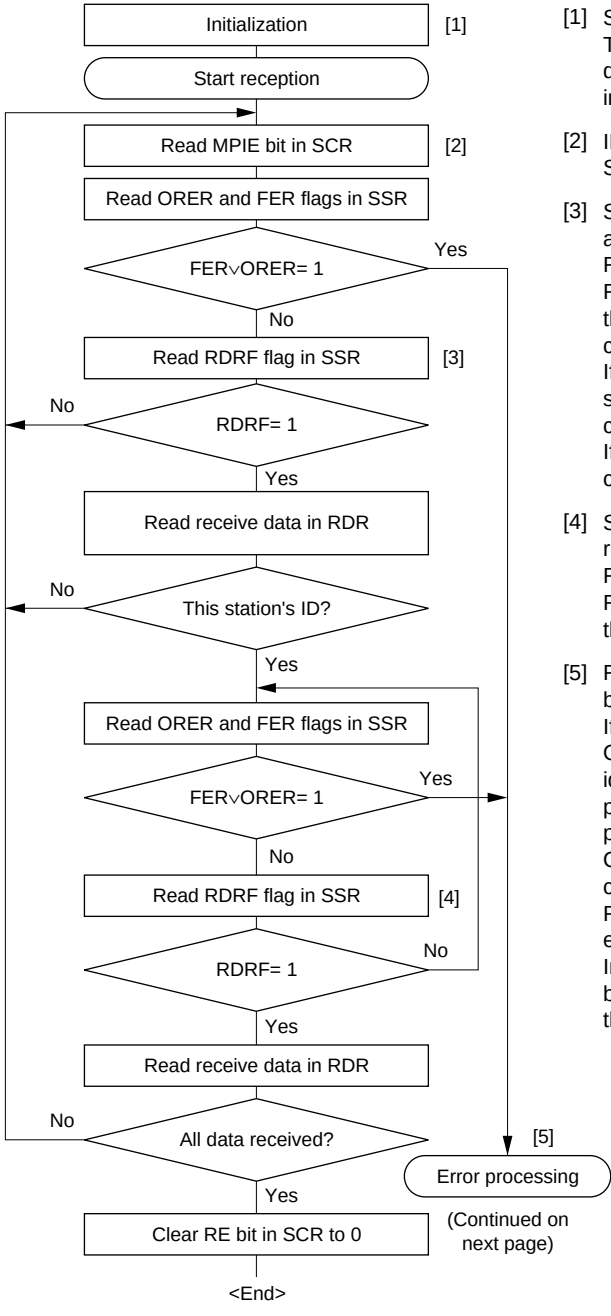



**Figure 14-11 Example of SCI Operation in Transmission (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

- Multiprocessor serial data reception

  Figure 14-12 shows a sample flowchart for multiprocessor serial reception.

  The following procedure should be used for multiprocessor serial data reception.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI status check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID.
If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0.
If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI status check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are all cleared to 0.
Reception cannot be resumed if either of these flags is set to 1.
In the case of a framing error, a break can be detected by reading the RxD pin value.

**Figure 14-12 Sample Multiprocessor Serial Reception Flowchart**

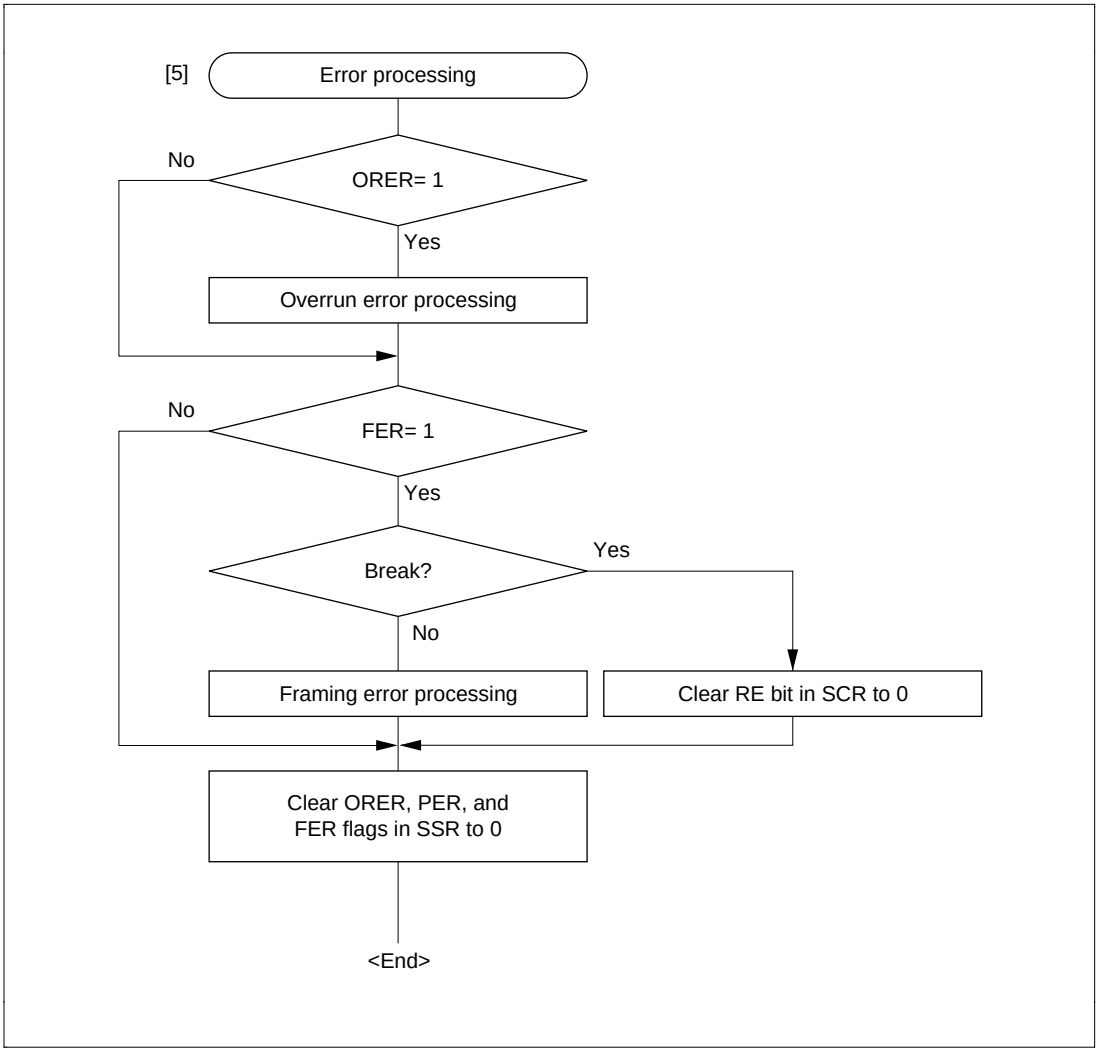



**Figure 14-12 Sample Multiprocessor Serial Reception Flowchart (cont)**

Figure 14-13 shows an example of SCI operation for multiprocessor format reception.

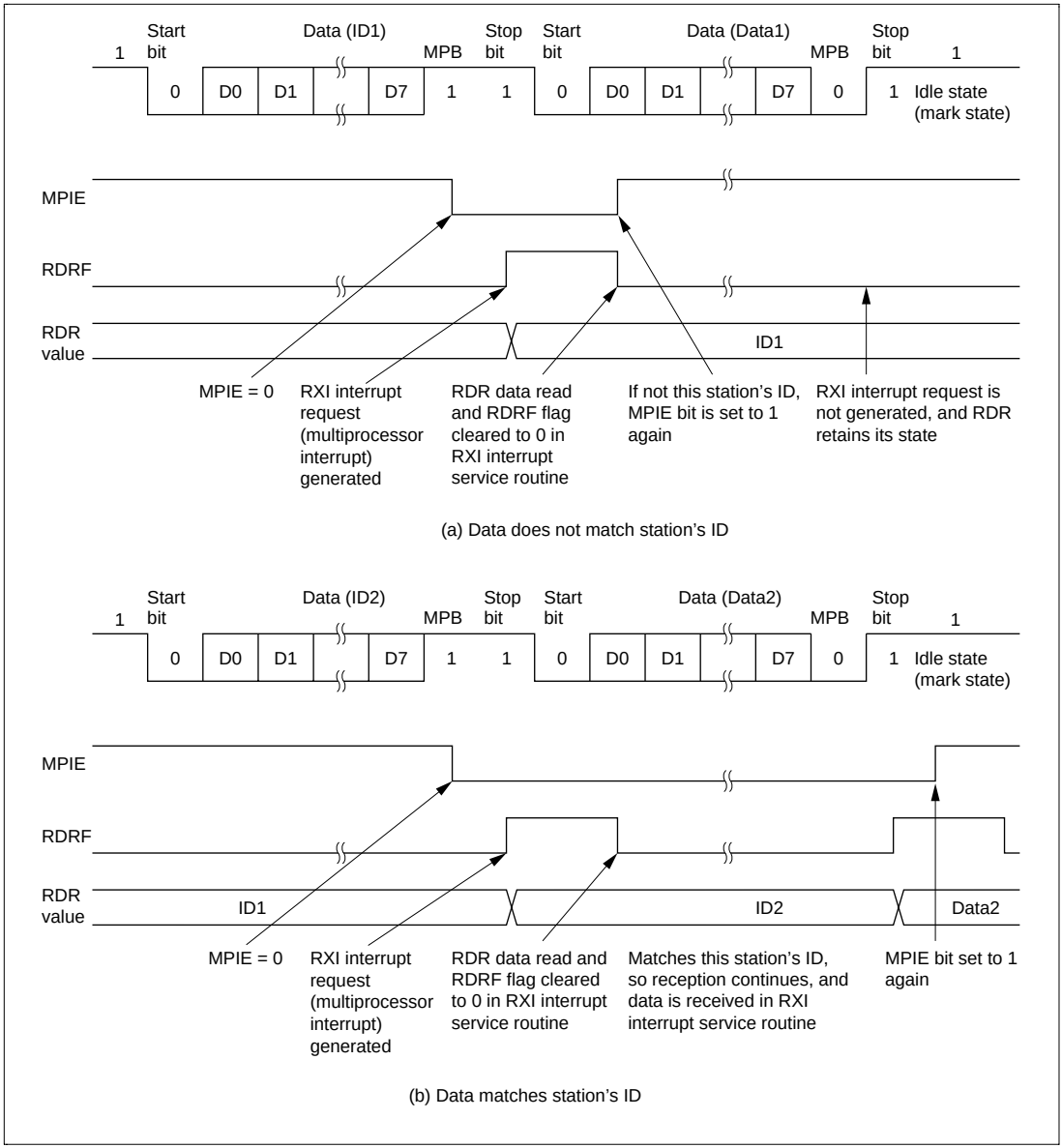



**Figure 14-13 Example of SCI Operation in Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

### 14.3.4 Operation in Clocked Synchronous Mode

In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for high-speed serial communication.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 14-14 shows the general format for clocked synchronous serial communication.

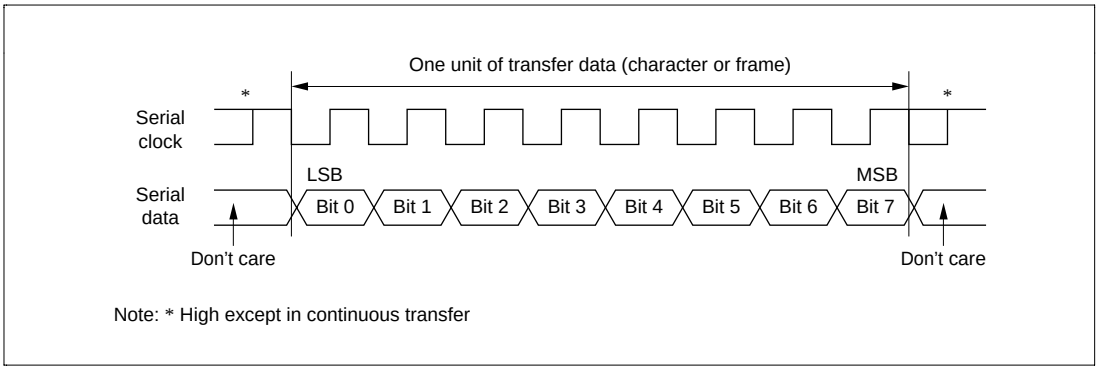



**Figure 14-14 Data Format in Synchronous Communication**

In clocked synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. Data confirmation is guaranteed at the rising edge of the serial clock.

In clocked serial communication, one character consists of data output starting with the LSB and ending with the MSB. After the MSB is output, the transmission line holds the MSB state.

In clocked synchronous mode, the SCI receives data in synchronization with the rising edge of the serial clock.

**Data Transfer Format:** A fixed 8-bit data format is used.

No parity or multiprocessor bits are added.

**Clock:** Either an internal clock generated by the on-chip baud rate generator or an external serial clock input at the SCK pin can be selected, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 14-9.

When the SCI is operated on an internal clock, the serial clock is output from the SCK pin.

Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. When only receive operations are performed, however, the serial clock is output until an overrun error occurs or the RE bit is cleared to 0. If you want to perform receive operations in units of one character, you should select an external clock as the clock source.

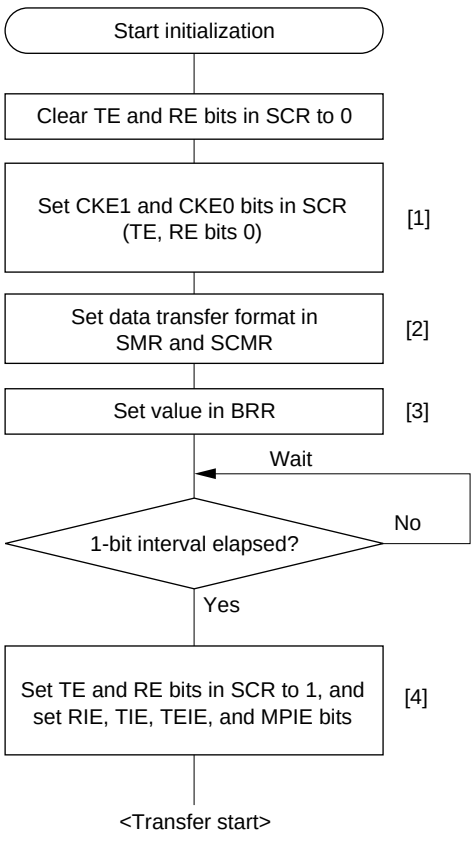
**Data Transfer Operations:**

- SCI initialization (clocked synchronous mode)

  Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

  When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

  Figure 14-15 shows a sample SCI initialization flowchart.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, TE and RE, to 0.

[2] Set the data transfer format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.
Setting the TE and RE bits enables the TxD and RxD pins to be used.

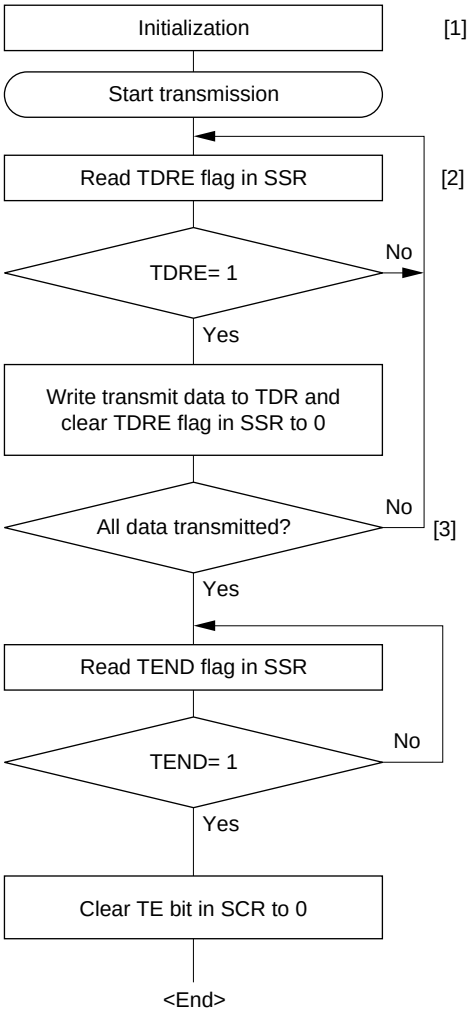
Note: In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

**Figure 14-15 Sample SCI Initialization Flowchart**

- Serial data transmission (clocked synchronous mode)

  Figure 14-16 shows a sample flowchart for serial transmission.

  The following procedure should be used for serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.
Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit data empty interrupt (TXI) request and data is written to TDR.

**Figure 14-16 Sample Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

When clock output mode has been set, the SCI outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

The serial transmit data is sent from the TxD pin starting with the LSB (bit 0) and ending with the MSB (bit 7).

[3] The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).

If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, and serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the MSB (bit 7) is sent, and the TxD pin maintains its state.

If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

[4] After completion of serial transmission, the SCK pin is fixed.

Figure 14-17 shows an example of SCI operation in transmission.

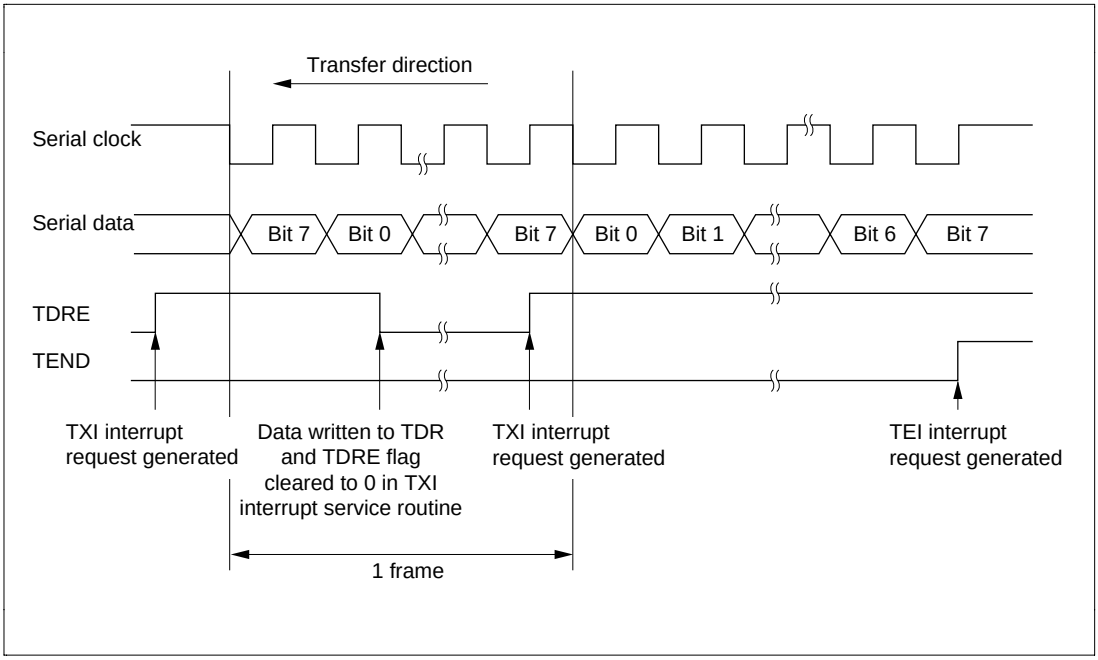


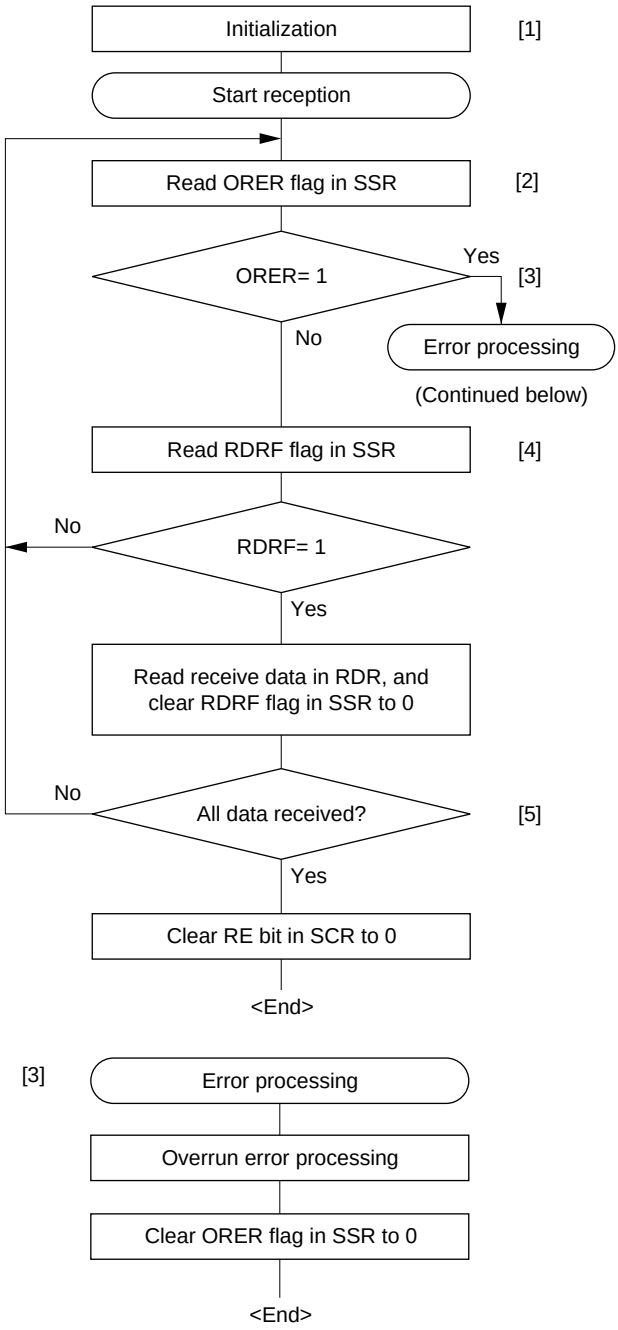

**Figure 14-17 Example of SCI Operation in Transmission**

- Serial data reception (clocked synchronous mode)
  Figure 14-18 shows a sample flowchart for serial reception.

  The following procedure should be used for serial data reception.

  When changing the operating mode from asynchronous to clocked synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0.

  The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR , and after performing the appropriate error processing, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0.
Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. The RDRF flag is cleared automatically when the DMAC or DTC is activated by a receive data full interrupt (RXI) request and the RDR value is read.

**Figure 14-18 Sample Serial Reception Flowchart**

In serial reception, the SCI operates as described below.

[1] The SCI performs internal initialization in synchronization with serial clock input or output.

[2] The received data is stored in RSR in LSB-to-MSB order.

After reception, the SCI checks whether the RDRF flag is 0 and the receive data can be transferred from RSR to RDR.

If this check is passed, the RDRF flag is set to 1, and the receive data is stored in RDR. If a receive error is detected in the error check, the operation is as shown in table 14-11.

Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

[3] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive data full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER flag changes to 1, a receive error interrupt (ERI) request is generated.

Figure 14-19 shows an example of SCI operation in reception.

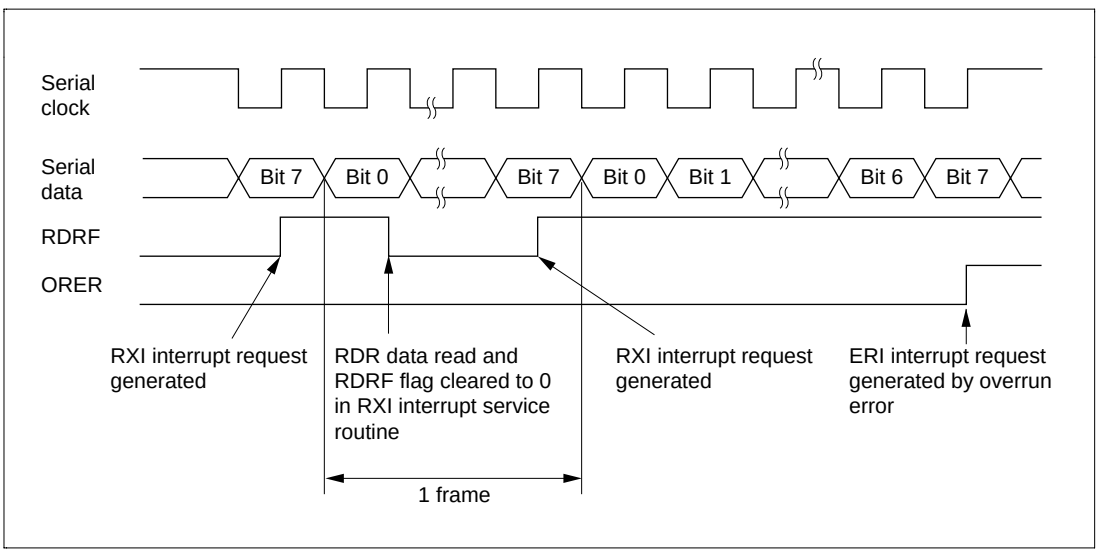
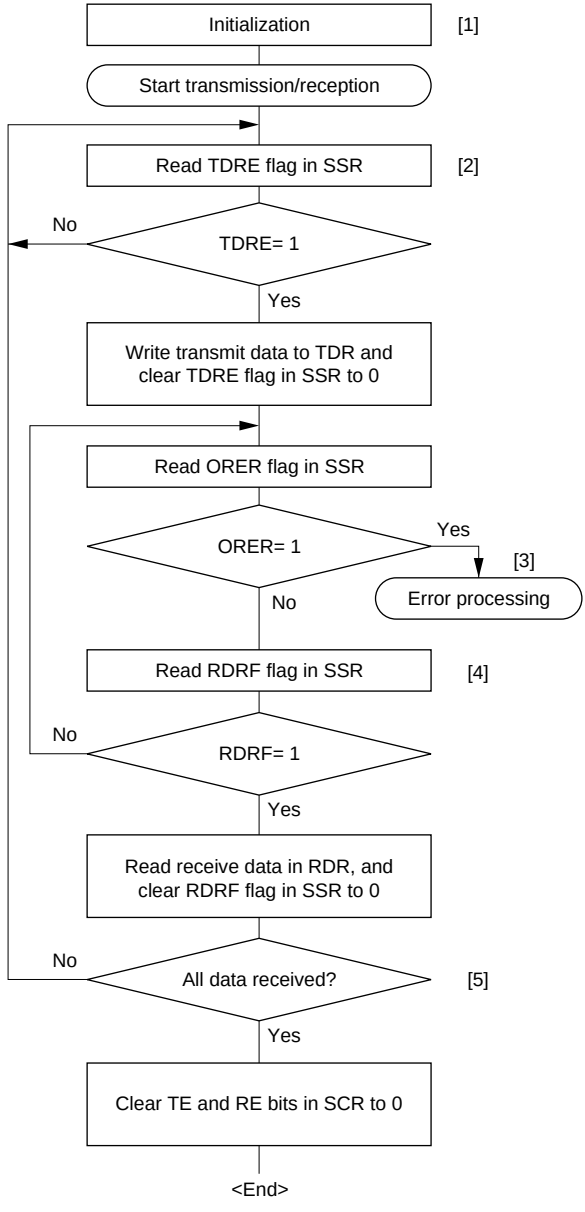



**Figure 14-19 Example of SCI Operation in Reception**

- Simultaneous serial data transmission and reception (clocked synchronous mode)

  Figure 14-20 shows a sample flowchart for simultaneous serial transmit and receive operations.

  The following procedure should be used for simultaneous serial data transmit and receive operations.




Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE and RE bits to 0, then set both these bits to 1 simultaneously.

[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.
Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR , and after performing the appropriate error processing, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/ reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit data empty interrupt (TXI) request and data is written to TDR. Also, the RDRF flag is cleared automatically when the DMAC or DTC is activated by a receive data full interrupt (RXI) request and the RDR value is read.

**Figure 14-20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations**

## 14.4 SCI Interrupts

The SCI has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request. Table 14-12 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in the SCR. Each kind of interrupt request is sent to the interrupt controller independently.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DMAC or DTC to perform data transfer. The TDRE flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. The DMAC and DTC cannot be activated by a TEI interrupt request.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DMAC or DTC to perform data transfer. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. The DMAC and DTC cannot be activated by an ERI interrupt request.

Also note that the DMAC cannot be activated by an SCI channel 2 interrupt.

**Table 14-12 SCI Interrupt Sources**

| Channel | Interrupt Source | Description | DTC Activation | DMAC Activation | Priority* |
|---|---|---|---|---|---|
| 0 | ERI | Interrupt due to receive error (ORER, FER, or PER) | Not possible | Not possible | High |
| | RXI | Interrupt due to receive data full state (RDRF) | Possible | Possible | |
| | TXI | Interrupt due to transmit data empty state (TDRE) | Possible | Possible | |
| | TEI | Interrupt due to transmission end (TEND) | Not possible | Not possible | |
| 1 | ERI | Interrupt due to receive error (ORER, FER, or PER) | Not possible | Not possible | |
| | RXI | Interrupt due to receive data full state (RDRF) | Possible | Possible | |
| | TXI | Interrupt due to transmit data empty state (TDRE) | Possible | Possible | |
| | TEI | Interrupt due to transmission end (TEND) | Not possible | Not possible | |
| 2 | ERI | Interrupt due to receive error (ORER, FER, or PER) | Not possible | Not possible | |
| | RXI | Interrupt due to receive data full state (RDRF) | Possible | Not possible | |
| | TXI | Interrupt due to transmit data empty state (TDRE) | Possible | Not possible | |
| | TEI | Interrupt due to transmission end (TEND) | Not possible | Not possible | Low |

Note: * This table shows the initial state immediately after a reset. Relative priorities among channels can be changed by means of the interrupt controller.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. The TEND flag is cleared at the same time as the TDRE flag. Consequently, if a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt may be accepted first, with the result that the TDRE and TEND flags are cleared. Note that the TEI interrupt will not be accepted in this case.

## 14.5 Usage Notes

The following points should be noted when using the SCI.

**Relation between Writes to TDR and the TDRE Flag**

The TDRE flag in SSR is a status flag that indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR regardless of the state of the TDRE flag. However, if new data is written to TDR when the TDRE flag is cleared to 0, the data stored in TDR will be lost since it has not yet been transferred to TSR. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to TDR.

**Operation when Multiple Receive Errors Occur Simultaneously**

If a number of receive errors occur at the same time, the state of the status flags in SSR is as shown in table 14-13. If there is an overrun error, data is not transferred from RSR to RDR, and the receive data is lost.

**Table 14-13 State of SSR Status Flags and Transfer of Receive Data**

| SSR Status Flags | | | | Receive Data Transfer | |
|---|---|---|---|---|---|
| RDRF | ORER | FER | PER | RSR to RDR | Receive Error Status |
| 1 | 1 | 0 | 0 | X | Overrun error |
| 0 | 0 | 1 | 0 | ◯ | Framing error |
| 0 | 0 | 0 | 1 | ◯ | Parity error |
| 1 | 1 | 1 | 0 | X | Overrun error + framing error |
| 1 | 1 | 0 | 1 | X | Overrun error + parity error |
| 0 | 0 | 1 | 1 | ◯ | Framing error + parity error |
| 1 | 1 | 1 | 1 | X | Overrun error + framing error + parity error |

Notes: ◯: Receive data is transferred from RSR to RDR.
X: Receive data is not transferred from RSR to RDR.

**Break Detection and Processing (Asynchronous Mode Only):** When framing error (FER) detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the parity error flag (PER) may also be set.

Note that, since the SCI continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

**Sending a Break (Asynchronous Mode Only):** The TxD pin has a dual function as an I/O port whose direction (input or output) is determined by DR and DDR. This can be used to send a break.

Between serial transmission initialization and setting of the TE bit to 1, the mark state is replaced by the value of DR (the pin does not function as the TxD pin until the TE bit is set to 1). Consequently, DDR and DR for the port corresponding to the TxD pin are first set to 1.

To send a break during serial transmission, first clear DR to 0, then clear the TE bit to 0.

When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

**Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only):**
Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

**Receive Data Sampling Timing and Reception Margin in Asynchronous Mode:**
In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the transfer rate.

In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the basic clock. This is illustrated in figure 14-21.

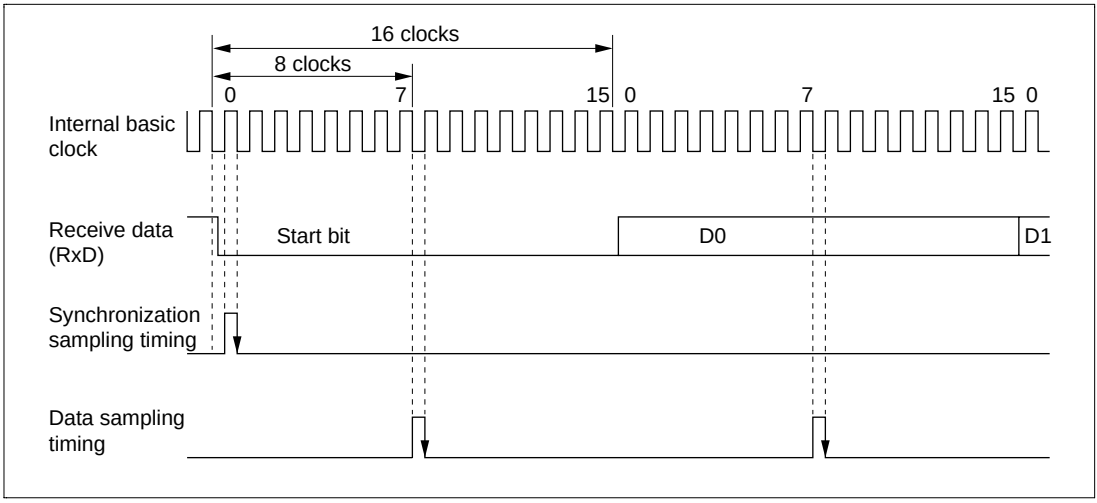



**Figure 14-21 Receive Data Sampling Timing in Asynchronous Mode**

Thus the reception margin in asynchronous mode is given by formula (1) below.

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}\,(1 + F) \right| \times 100\% \quad \ldots\ldots\ldots \text{Formula (1)}$$

Where M : Reception margin (%)
N : Ratio of bit rate to clock (N = 16)
D : Clock duty (D = 0 to 1.0)
L : Frame length (L = 9 to 12)
F : Absolute value of clock rate deviation

Assuming values of F = 0 and D = 0.5 in formula (1), a reception margin of 46.875% is given by formula (2) below.

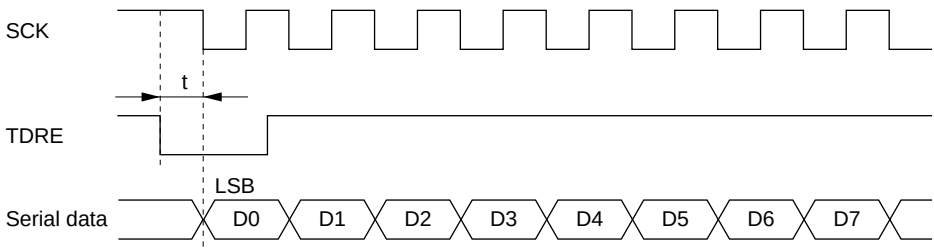
When D = 0.5 and F = 0,

$$M = (0.5 - \frac{1}{2 \times 16}) \times 100\%$$
$$= 46.875\% \quad \ldots\ldots\ldots \text{Formula (2)}$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

**Restrictions on Use of DMAC or DTC**

- When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5 ø clock cycles after TDR is updated by the DMAC or DTC. Misoperation may occur if the transmit clock is input within 4 ø clocks after TDR is updated. (Figure 14-22)
- When RDR is read by the DMAC or DTC, be sure to set the activation source to the relevant SCI reception end interrupt (RXI).




Note: When operating on an external clock, set t >4 clocks.

**Figure 14-22 Example of Clocked Synchronous Transmission by DTC**

# Section 15 Smart Card Interface

## 15.1 Overview

SCI supports an IC card (Smart Card) interface conforming to ISO/IEC 7816-3 (Identification Card) as a serial communication interface extension function.

Switching between the normal serial communication interface and the Smart Card interface is carried out by means of a register setting.

### 15.1.1 Features

Features of the Smart Card interface supported by the H8S/2655 Series are as follows.

- Asynchronous mode
  - Data length: 8 bits
  - Parity bit generation and checking
  - Transmission of error signal (parity error) in receive mode
  - Error signal d648etection and automatic data retransmission in transmit mode
  - Direct convention and inverse convention both supported
- On-chip baud rate generator allows any bit rate to be selected
- Three interrupt sources
  - Three interrupt sources (transmit data empty, receive data full, and transmit/receive error) that can issue requests independently
  - The transmit data empty interrupt and receive data full interrupt can activate the DMA controller (DMAC) or data transfer controller (DTC) to execute data transfer

### 15.1.2 Block Diagram

Figure 15-1 shows a block diagram of the Smart Card interface.

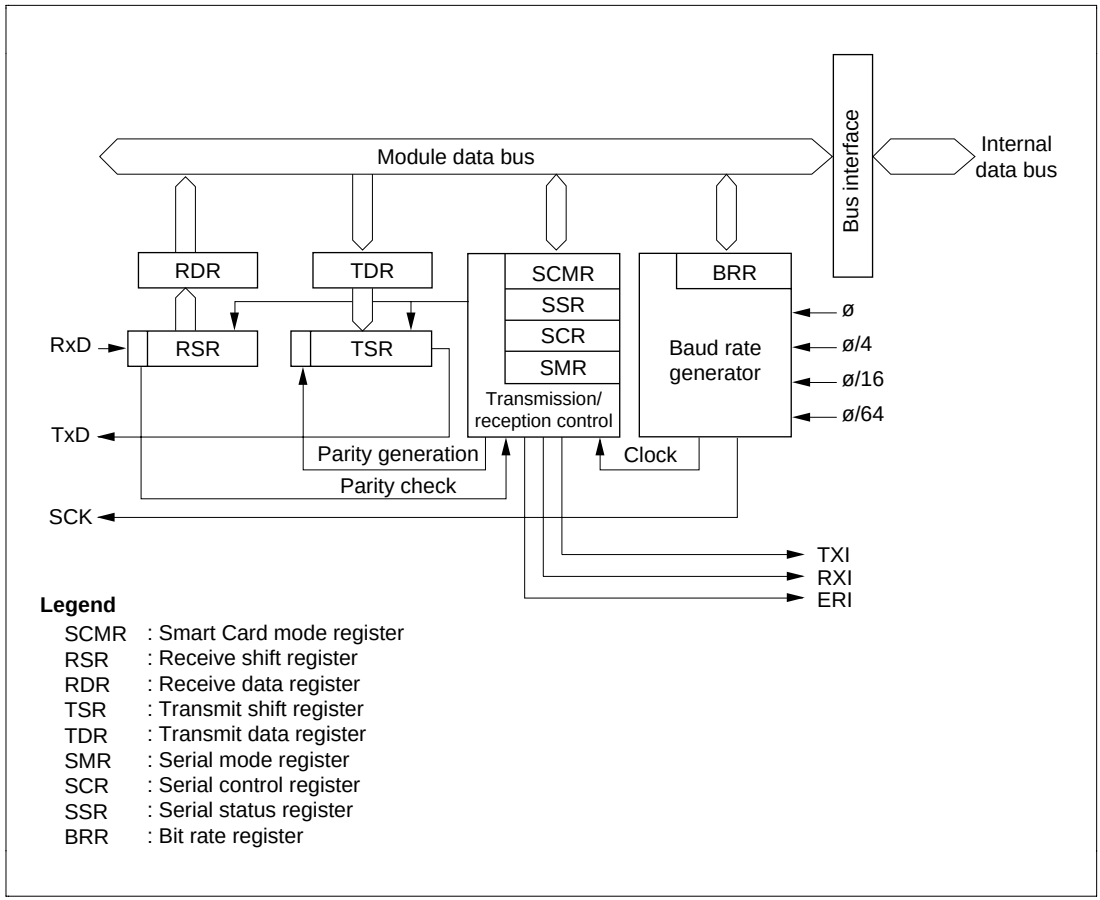



**Figure 15-1 Block Diagram of Smart Card Interface**

### 15.1.3 Pin Configuration

Table 15-1 shows the Smart Card interface pin configuration.

**Table 15-1 Smart Card Interface Pins**

| Channel | Pin Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Serial clock pin 0 | $SCK_0$ | I/O | SCI0 clock input/output |
| | Receive data pin 0 | $RxD_0$ | Input | SCI0 receive data input |
| | Transmit data pin 0 | $TxD_0$ | Output | SCI0 transmit data output |
| 1 | Serial clock pin 1 | $SCK_1$ | I/O | SCI1 clock input/output |
| | Receive data pin 1 | $RxD_1$ | Input | SCI1 receive data input |
| | Transmit data pin 1 | $TxD_1$ | Output | SCI1 transmit data output |
| 2 | Serial clock pin 2 | $SCK_2$ | I/O | SCI2 clock input/output |
| | Receive data pin 2 | $RxD_2$ | Input | SCI2 receive data input |
| | Transmit data pin 2 | $TxD_2$ | Output | SCI2 transmit data output |

### 15.1.4 Register Configuration

Table 15-2 shows the registers used by the Smart Card interface. Details of SMR, BRR, SCR, TDR, RDR, and MSTPCR are the same as for the normal SCI function: see the register descriptions in section 14, Serial Communication Interface.

**Table 15-2 Smart Card Interface Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Serial mode register 0 | SMR0 | R/W | H'00 | H'FF78 |
| | Bit rate register 0 | BRR0 | R/W | H'FF | H'FF79 |
| | Serial control register 0 | SCR0 | R/W | H'00 | H'FF7A |
| | Transmit data register 0 | TDR0 | R/W | H'FF | H'FF7B |
| | Serial status register 0 | SSR0 | R/(W)*[2] | H'84 | H'FF7C |
| | Receive data register 0 | RDR0 | R | H'00 | H'FF7D |
| | Smart card mode register 0 | SCMR0 | R/W | H'F2 | H'FF7E |
| 1 | Serial mode register 1 | SMR1 | R/W | H'00 | H'FF80 |
| | Bit rate register 1 | BRR1 | R/W | H'FF | H'FF81 |
| | Serial control register 1 | SCR1 | R/W | H'00 | H'FF82 |
| | Transmit data register 1 | TDR1 | R/W | H'FF | H'FF83 |
| | Serial status register 1 | SSR1 | R/(W)*[2] | H'84 | H'FF84 |
| | Receive data register 1 | RDR1 | R | H'00 | H'FF85 |
| | Smart card mode register 1 | SCMR1 | R/W | H'F2 | H'FF86 |
| 2 | Serial mode register 2 | SMR2 | R/W | H'00 | H'FF88 |
| | Bit rate register 2 | BRR2 | R/W | H'FF | H'FF89 |
| | Serial control register 2 | SCR2 | R/W | H'00 | H'FF8A |
| | Transmit data register 2 | TDR2 | R/W | H'FF | H'FF8B |
| | Serial status register 2 | SSR2 | R/(W)*[2] | H'84 | H'FF8C |
| | Receive data register 2 | RDR2 | R | H'00 | H'FF8D |
| | Smart card mode register 2 | SCMR2 | R/W | H'F2 | H'FF8E |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 15.2 Register Descriptions

Registers added with the Smart Card interface and bits for which the function changes are described here.

### 15.2.1 Smart Card Mode Register (SCMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | — | R/W |

SCMR is an 8-bit readable/writable register that selects the Smart Card interface function.

SCMR is initialized to H'F2 by a reset, and in standby mode or module stop mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format.

**Bit 3**

| SDIR | Description | |
|---|---|---|
| 0 | TDR contents are transmitted LSB-first | (Initial value) |
| | Receive data is stored in RDR LSB-first | |
| 1 | TDR contents are transmitted MSB-first | |
| | Receive data is stored in RDR MSB-first | |

**Bit 2—Smart Card Data Invert (SINV):** Specifies inversion of the data logic level. This function is used together with the SDIR bit for communication with an inverse convention card. The SINV bit does not affect the logic level of the parity bit. For parity-related setting procedures, see section 15.3.4, Register Settings.

**Bit 2**

| SINV | Description | |
|---|---|---|
| 0 | TDR contents are transmitted as they are | (Initial value) |
| | Receive data is stored as it is in RDR | |
| 1 | TDR contents are inverted before being transmitted | |
| | Receive data is stored in inverted form in RDR | |

**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** Enables or disables the Smart Card interface function.

**Bit 0**

| SMIF | Description | |
|---|---|---|
| 0 | Smart Card interface function is disabled | (Initial value) |
| 1 | Smart Card interface function is enabled | |

### 15.2.2 Serial Status Register (SSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Note: * Only 0 can be written to bits 7 to 3, to clear these flags.

Bit 4 of SSR has a different function in Smart Card interface mode. Coupled with this, the setting conditions for bit 2, TEND, are also different.

**Bits 7 to 5—**Operate in the same way as for the normal SCI. For details, see section 14.2.7, Serial Status Register (SSR).

**Bit 4—Error Signal Status (ERS):** In Smart Card interface mode, bit 4 indicates the status of the error signal sent back from the receiving end in transmission. Framing errors are not detected in Smart Card interface mode.

**Bit 4**

| ERS | Description | |
|---|---|---|
| 0 | [Clearing condition] | (Initial value) |
| | • Upon reset, and in standby mode or module stop mode | |
| | • When 0 is written to ERS after reading ERS = 1 | |
| 1 | [Setting condition] | |
| | When the low level of the error signal is sampled | |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its previous state.

**Bits 3 to 0—**Operate in the same way as for the normal SCI. For details, see section 14.2.7, Serial Status Register (SSR).

However, the setting conditions for the TEND bit, are as shown below.

**Bit 2**

| TEND | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When data is written to TDR by the DMAC or DTC |
| 1 | [Setting conditions]<br>• Upon reset, and in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 2.5 etu after transmission of a 1-byte serial character |

Note: etu: Elementary Time Unit (time for transfer of 1 bit)

### 15.2.3 Serial Mode Register (SMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Set value* | : | GM | 0 | 1 | O/$\overline{E}$ | 1 | 0 | CKS1 | CKS0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * When the smart card interface is used, be sure to make the 0 or 1 setting shown for bits 6, 5, 3, and 2.

The function of bit 7 of SMR changes in smart card interface mode.

**Bit 7—GSM Mode (GM):** Sets the smart card interface function to GSM mode.

This bit is cleared to 0 when the normal smart card interface is used. In GSM mode, this bit is set to 1, the timing of setting of the TEND flag that indicates transmission completion is advanced and clock output control mode addition is performed. The contents of the clock output control mode addition are specified by bits 1 and 0 of the serial control register (SCR).

**Bit 7**

| GM | Description |
|---|---|
| 0 | Normal smart card interface mode operation (Initial value)<br>• TEND flag generation 12.5 etu after beginning of start bit<br>• Clock output ON/OFF control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generation 11.0 etu after beginning of start bit<br>• High/low fixing control possible in addition to clock output ON/OFF control (set by SCR) |

Note: etu: Elementary time unit (time for transfer of 1 bit)

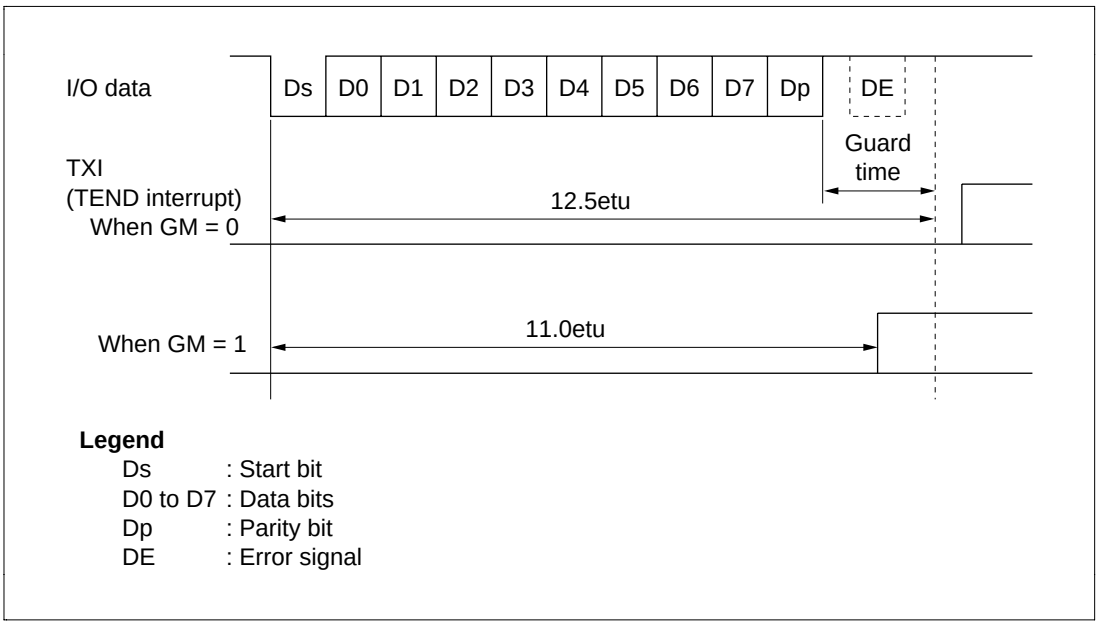



**Figure 15-2 TEND Flag Generation Timing in Transmission Operation**

### 15.2.4 Serial Control Register (SCR)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

In smart card interface mode, the function of bits 1 and 0 of SCR changes when bit 7 of the serial mode register (SMR) is set to 1.

**Bits 7 to 2—Operate in the dame way as for the normal SCI.**

For details, see section 14.2.6, Serial Control Register (SCR).

**Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0):** These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin.

In smart card interface mode, in addition to the normal switching between clock output enabling and disabling, the clock output can be specified as to be fixed high or low.

| SCMR | SMR | SCR Setting | | |
|---|---|---|---|---|
| **SMIF** | **C/$\overline{A}$, GM** | **CKE1** | **CKE0** | **SCK Pin Function** |
| 0 | See the SCI | | | |
| 1 | 0 | 0 | 0 | Operates as port I/O pin |
| 1 | 0 | 0 | 1 | Outputs clock as SCK output pin |
| 1 | 1 | 0 | 0 | Operates as SCK output pin, with output fixed low |
| 1 | 1 | 0 | 1 | Outputs clock as SCK output pin |
| 1 | 1 | 1 | 0 | Operates as SCK output pin, with output fixed high |
| 1 | 1 | 1 | 1 | Outputs clock as SCK output pin |

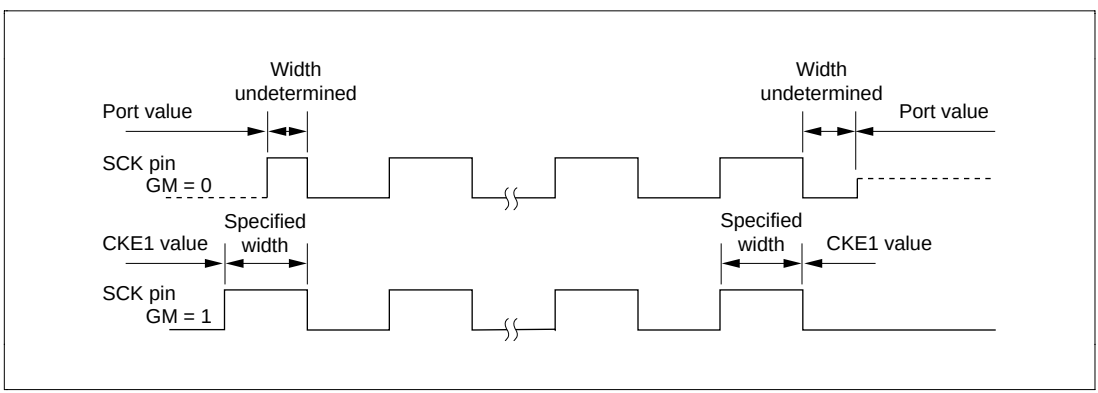



**Figure 15-3 Clock Output Waveform Control**

## 15.3 Operation

### 15.3.1 Overview

The main functions of the Smart Card interface are as follows.

- One frame consists of 8-bit data plus a parity bit.
- In transmission, a guard time of at least 2 etu (Elementary Time Unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
- If a parity error is detected during reception, a low error signal level is output for one etu period, 10.5 etu after the start bit.
- If the error signal is sampled during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer.
- Only asynchronous communication is supported; there is no clocked synchronous communication function.

### 15.3.2 Pin Connections

Figure 15-4 shows a schematic diagram of Smart Card interface related pin connections.

In communication with an IC card, since both transmission and reception are carried out on a single data transmission line, the TxD pin and RxD pin should be connected with the LSI pin. The data transmission line should be pulled up to the $V_{CC}$ power supply with a resistor.

When the clock generated on the Smart Card interface is used by an IC card, the SCK pin output is input to the CLK pin of the IC card. No connection is needed if the IC card uses an internal clock.

LSI port output is used as the reset signal.

Other pins must normally be connected to the power supply or ground.

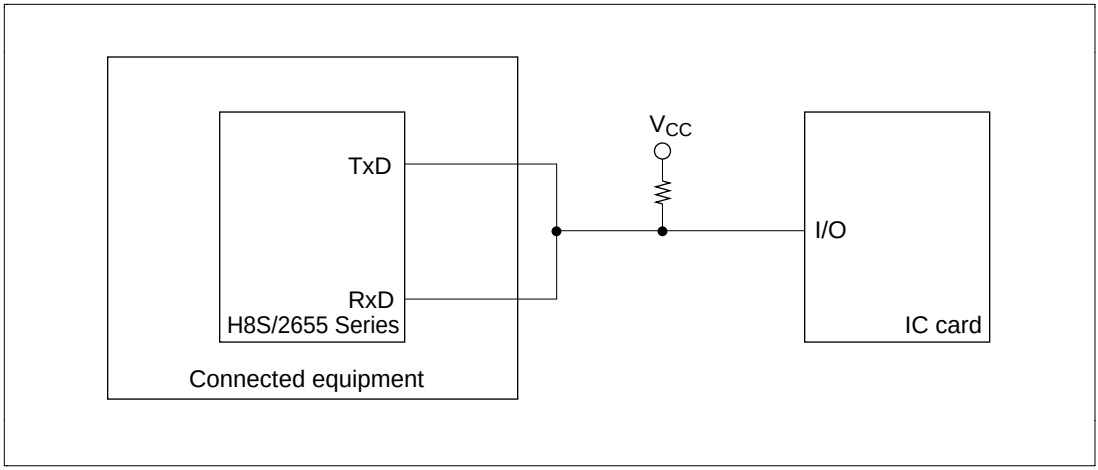



**Figure 15-4 Schematic Diagram of Smart Card Interface Pin Connections**

Note: If an IC card is not connected, and the TE and RE bits are both set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out.

### 15.3.3 Data Format

Figure 15-5 shows the Smart Card interface data format. In reception in this mode, a parity check is carried out on each frame, and if an error is detected an error signal is sent back to the transmitting end, and retransmission of the data is requested. If an error signal is sampled during transmission, the same data is retransmitted.

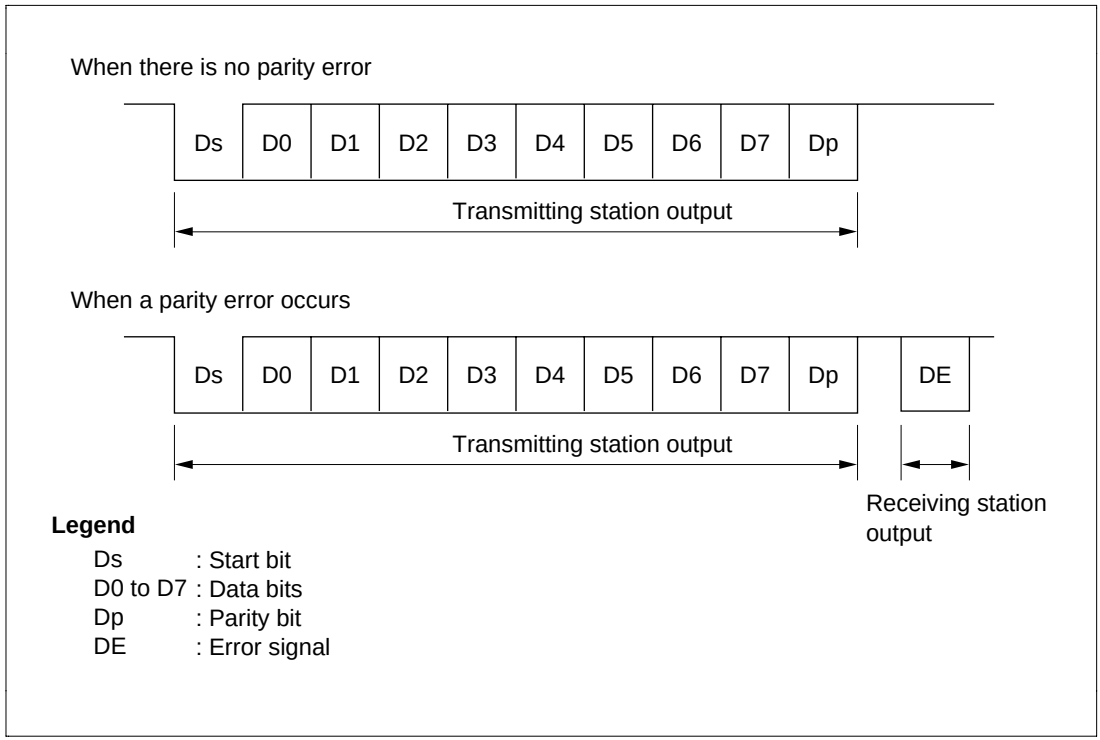



**Figure 15-5 Smart Card Interface Data Format**

The operation sequence is as follows.

[1] When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.

[2] The transmitting station starts transfer of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).

[3] With the Smart Card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.

[4] The receiving station carries out a parity check.

If there is no parity error and the data is received normally, the receiving station waits for reception of the next data.

If a parity error occurs, however, the receiving station outputs an error signal (DE, low-level) to request retransmission of the data. After outputting the error signal for the prescribed length of time, the receiving station places the signal line in the high-impedance state again. The signal line is pulled high again by a pull-up resistor.

[5] If the transmitting station does not receive an error signal, it proceeds to transmit the next data frame.

If it does receive an error signal, however, it returns to step [2] and retransmits the erroneous data.

### 15.3.4 Register Settings

Table 15-3 shows a bit map of the registers used by the smart card interface.

Bits indicated as 0 or 1 must be set to the value shown. The setting of other bits is described below.

**Table 15-3 Smart Card Interface Register Settings**

| | Bit | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Register** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| SMR | GM | 0 | 1 | O/$\overline{E}$ | 1 | 0 | CKS1 | CKS0 |
| BRR | BRR7 | BRR6 | BRR5 | BRR4 | BRR3 | BRR2 | BRR1 | BRR0 |
| SCR | TIE | RIE | TE | RE | 0 | 0 | CKE1 | CKE0 |
| TDR | TDR7 | TDR6 | TDR5 | TDR4 | TDR3 | TDR2 | TDR1 | TDR0 |
| SSR | TDRE | RDRF | ORER | ERS | PER | TEND | 0 | 0 |
| RDR | RDR7 | RDR6 | RDR5 | RDR4 | RDR3 | RDR2 | RDR1 | RDR0 |
| SCMR | — | — | — | — | SDIR | SINV | — | SMIF |

Notes: — : Unused bit.

**SMR Setting:** The GM bit controls the TEND flag set timing and clock output. The O/$\overline{E}$ bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

Bits CKS1 and CKS0 select the clock source of the on-chip baud rate generator. See section 15.3.5, Clock.

**BRR Setting:** BRR is used to set the bit rate. See section 15.3.5, Clock, for the method of calculating the value to be set.

**SCR Setting:** The function of the TIE, RIE, TE, and RE bits is the same as for the normal SCI. For details, see section 14, Serial Communication Interface.

Bits CKE1 and CKE0 specify the clock output. When the GM bit in SMR is cleared to 0, set these bits to B'00 if a clock is not to be output, or to B'01 if a clock is to be output. When the GM bit in SMR is set to 1, clock output is performed. The clock output can also be fixed high or low.

**Smart Card Mode Register (SCMR) Setting:**

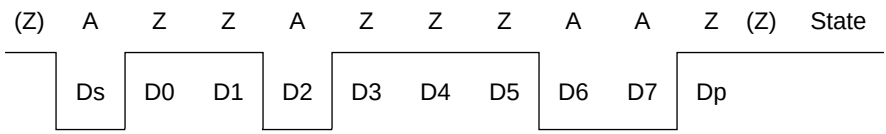
The SDIR bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if  of the inverse convention type.

The SINV bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if  of the inverse convention type.

The SMIF bit is set to 1 in the case of the Smart Card interface.

Examples of register settings and the waveform of the start character are shown below for the two types of IC card (direct convention and inverse convention).

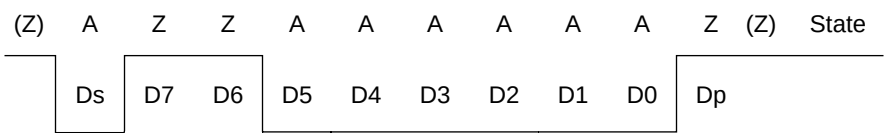
- Direct convention (SDIR = SINV = O/$\overline{E}$ = 0)




With the direct convention type, the logic 1 level corresponds to state Z and the logic 0 level to state A, and transfer is performed in LSB-first order. The start character data above is H'3B.

The parity bit is 1 since even parity is stipulated for the Smart Card.

- Inverse convention (SDIR = SINV = O/$\overline{E}$ = 1)




With the inverse convention type, the logic 1 level corresponds to state A and the logic 0 level to state Z, and transfer is performed in MSB-first order. The start character data above is H'3F.

The parity bit is 0, corresponding to state Z, since even parity is stipulated for the Smart Card.

With the H8S/2655 Series, inversion specified by the SINV bit applies only to the data bits, D7 to D0. For parity bit inversion, the O/$\overline{E}$ bit in SMR is set to odd parity mode (the same applies to both transmission and reception).

### 15.3.5 Clock

Only an internal clock generated by the on-chip baud rate generator can be used as the transmit/receive clock for the smart card interface. The bit rate is set with BRR and the CKS1 and CKS0 bits in SMR. The formula for calculating the bit rate is as shown below. Table 15-5 shows some sample bit rates.

If clock output is selected by setting CKE0 to 1, a clock with a frequency of 372 times the bit rate is output from the SCK pin.

$$B = \frac{\phi}{1488 \times 2^{2n-1} \times (N + 1)} \times 10^6$$

Where: N = Value set in BRR ($0 \leq N \leq 255$)
B = Bit rate (bit/s)
ø = Operating frequency (MHz)
n = See table 15-4

**Table 15-4 Correspondence between n and CKS1, CKS0**

| n | CKS1 | CKS0 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | | 1 |
| 2 | 1 | 0 |
| 3 | | 1 |

**Table 15-5 Examples of Bit Rate B (bit/s) for Various BRR Settings (When n = 0)**

| | ø (MHz) | | | | | |
|---|---|---|---|---|---|---|
| **N** | **10.00** | **10.714** | **13.00** | **14.285** | **16.00** | **18.00** |
| 0 | 13441 | 14400 | 17473 | 19200 | 21505 | 24194 |
| 1 | 6720 | 7200 | 8737 | 9600 | 10753 | 12097 |
| 2 | 4480 | 4800 | 5824 | 6400 | 7168 | 8065 |

Note: Bit rates are rounded to the nearest whole number.

The method of calculating the value to be set in the bit rate register (BRR) from the operating frequency and bit rate, on the other hand, is shown below. N is an integer, $0 \leq N \leq 255$, and the smaller error is specified.

$$N = \frac{\phi}{1488 \times 2^{2n-1} \times B} \times 10^6 - 1$$

**Table 15-6 Examples of BRR Settings for Bit Rate B (bit/s) (When n = 0)**

| | ø (MHz) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **7.1424** | | **10.00** | | **10.7136** | | **13.00** | | **14.2848** | | **16.00** | | **18.00** | |
| **bit/s** | **N** | **Error** | **N** | **Error** | **N** | **Error** | **N** | **Error** | **N** | **Error** | **N** | **Error** | **N** | **Error** |
| 9600 | 0 | 0.00 | 1 | 30 | 1 | 25 | 1 | 8.99 | 1 | 0.00 | 1 | 12.01 | 2 | 15.99 |

**Table 15-7 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)**

| **ø (MHz)** | **Maximum Bit Rate (bit/s)** | **N** | **n** |
|---|---|---|---|
| 7.1424 | 9600 | 0 | 0 |
| 10.00 | 13441 | 0 | 0 |
| 10.7136 | 14400 | 0 | 0 |
| 13.00 | 17473 | 0 | 0 |
| 14.2848 | 19200 | 0 | 0 |
| 16.00 | 21505 | 0 | 0 |
| 18.00 | 24194 | 0 | 0 |
| 20.00 | 26882 | 0 | 0 |

The bit rate error is given by the following formula:

$$\text{Error (\%)} = \left( \frac{\phi}{1488 \times 2^{2n-1} \times B \times (N + 1)} \times 10^6 - 1 \right) \times 100$$

### 15.3.6 Data Transfer Operations

**Initialization:** Before transmitting and receiving data, initialize the SCI as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa.

[1] Clear the TE and RE bits in SCR to 0.

[2] Clear the error flags ERS, PER, and ORER in SSR to 0.

[3] Set the O/$\overline{E}$ bit and CKS1 and CKS0 bits in SMR. Clear the C/$\overline{A}$, CHR, and MP bits to 0, and set the STOP and PE bits to 1.

[4] Set the SMIF, SDIR, and SINV bits in SCMR.
When the SMIF bit is set to 1, the TxD and RxD pins are both switched from ports to SCI pins, and are placed in the high-impedance state.

[5] Set the value corresponding to the bit rate in BRR.

[6] Set the CKE0 bit in SCR. Clear the TIE, RIE, TE, RE, MPIE, TEIE and CKE1 bits to 0.
If the CKE0 bit is set to 1, the clock is output from the SCK pin.

[7] Wait at least one bit interval, then set the TIE, RIE, TE, and RE bits in SCR. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.

**Serial Data Transmission:** As data transmission in smart card mode involves error signal sampling and retransmission processing, the processing procedure is different from that for the normal SCI. Figure 15-6 shows an example of the transmission processing flow.

[1] Perform Smart Card interface mode initialization as described above in Initialization.

[2] Check that the ERS error flag in SSR is cleared to 0.

[3] Repeat steps [2] and [3] until it can be confirmed that the TEND flag in SSR is set to 1.

[4] Write the transmit data to TDR, clear the TDRE flag to 0, and perform the transmit operation. The TEND flag is cleared to 0.

[5] When transmitting data continuously, go back to step [2].

[6] To end transmission, clear the TE bit to 0.

With the above processing, interrupt servicing or data transfer by the DMAC or DTC is possible.

If transmission ends and the TEND flag is set to 1 while the TIE bit is set to 1 and interrupt requests are enabled, a transmit data empty interrupt (TXI) request will be generated. If an error occurs in transmission and the ERS flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a transfer error interrupt (ERI) request will be generated.

The timing for setting the TEND flag depends on the value of the GM bit in SMR. The TEND flag set timing is shown in figure 15-6.

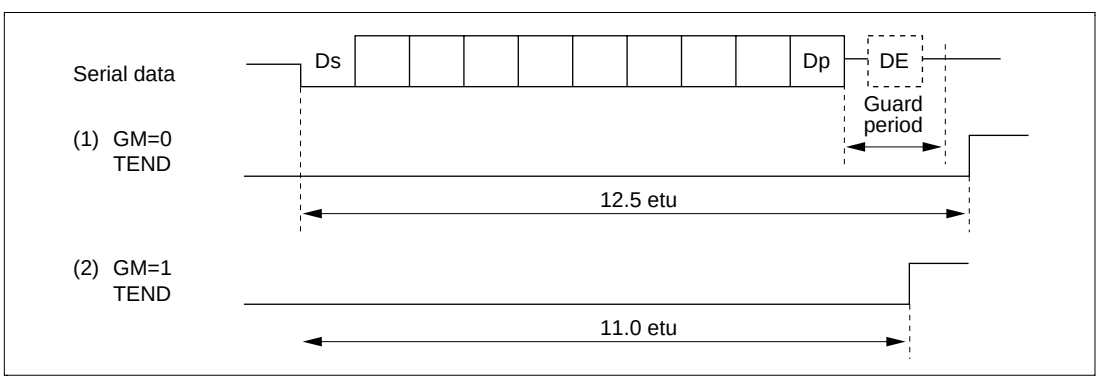



**Figure 15-6 TEND Flag Set Timing**

If the DMAC or DTC is activated by a TXI request, the number of bytes set in the DMAC or DTC can be transmitted automatically, including automatic retransmission.

For details, see Interrupt Operations and Data Transfer Operation by DMAC or DTC below.

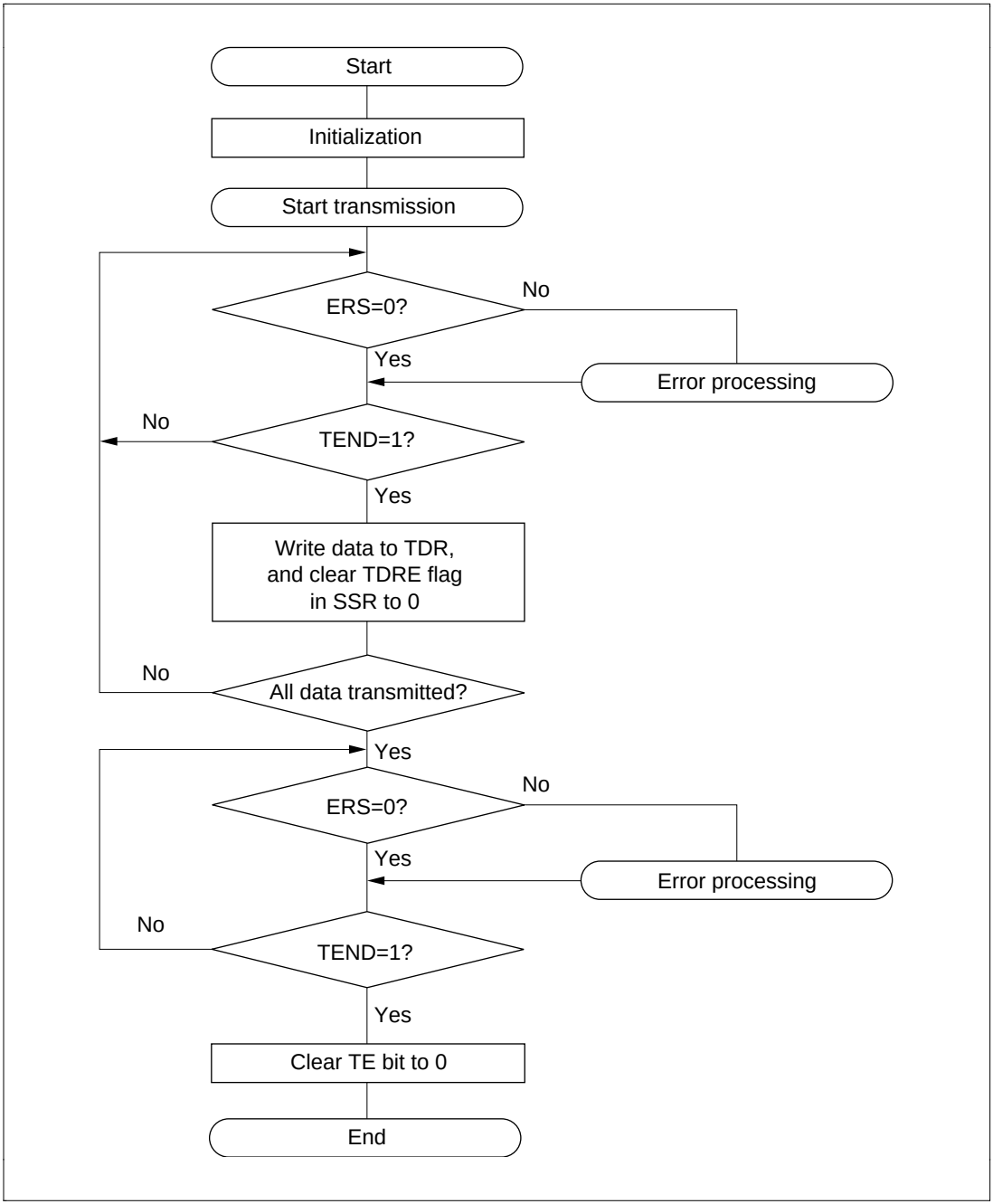



**Figure 15-7 Example of Transmission Processing Flow**

**Serial Data Reception:** Data reception in Smart Card mode uses the same processing procedure as for the normal SCI. Figure 15-8 shows an example of the transmission processing flow.

[1] Perform Smart Card interface mode initialization as described above in Initialization.

[2] Check that the ORER flag and PER flag in SSR are cleared to 0. If either is set, perform the appropriate receive error processing, then clear both the ORER and the PER flag to 0.

[3] Repeat steps [2] and [3] until it can be confirmed that the RDRF flag is set to 1.

[4] Read the receive data from RDR.

[5] When receiving data continuously, clear the RDRF flag to 0 and go back to step [2].

[6] To end reception, clear the RE bit to 0.

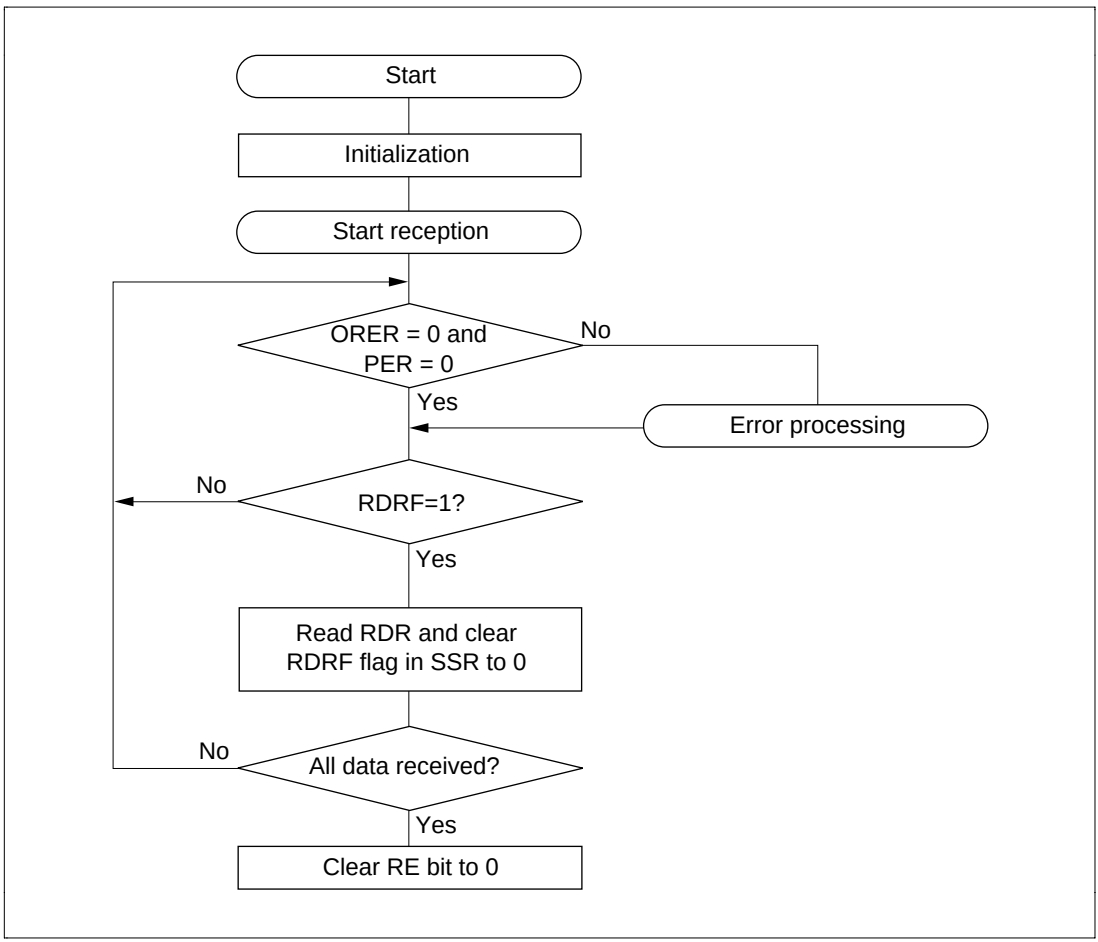



**Figure 15-8 Example of Reception Processing Flow**

With the above processing, interrupt servicing or data transfer by the DMAC or DTC is possible.

If reception ends and the RDRF flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a receive data full interrupt (RXI) request will be generated. If an error occurs in reception and either the ORER flag or the PER flag is set to 1, a transfer error interrupt (ERI) request will be generated.

If the DMAC or DTC is activated by an RXI request, the receive data in which the error occurred is skipped, and only the number of bytes of receive data set in the DMAC or DTC are transferred.

For details, see Interrupt Operation and Data Transfer Operation by DMAC or DTC below.

If a parity error occurs during reception and the PER is set to 1, the received data is still transferred to RDR, and therefore this data can be read.

**Mode Switching Operation:** When switching from receive mode to transmit mode, first confirm that the receive operation has been completed, then start from initialization, clearing RE bit to 0 and setting TE bit to 1. The RDRF flag or the PER and ORER flags can be used to check that the receive operation has been completed.

When switching from transmit mode to receive mode, first confirm that the transmit operation has been completed, then start from initialization, clearing TE bit to 0 and setting RE bit to 1. The TEND flag can be used to check that the transmit operation has been completed.

**Fixing Clock Output Level:** When the GSM bit in SMR is set to 1, the clock output level can be fixed with bits CKE1 and CKE0 in SCR. At this time, the minimum clock pulse width can be made the specified width.

Figure 15-9 shows the timing for fixing the clock output level. In this example, GSM is set to 1, CKE1 is cleared to 0, and the CKE0 bit is controlled.

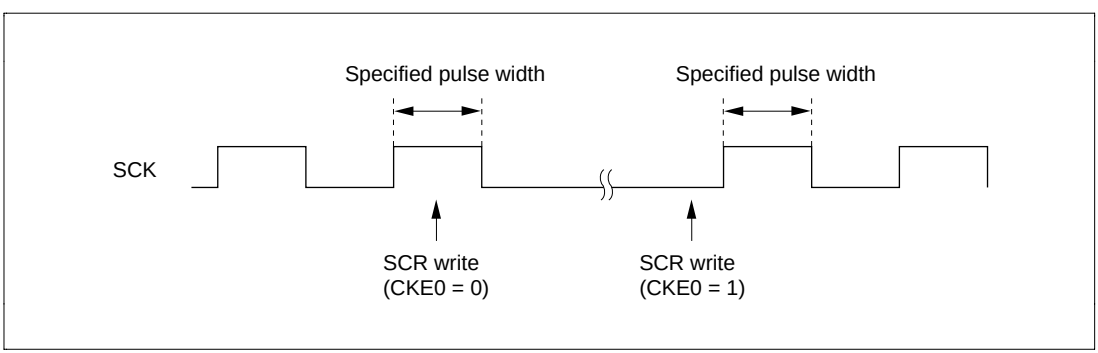



**Figure 15-9 Timing for Fixing Clock Output Level**

**Interrupt Operation:** There are three interrupt sources in smart card interface mode: transmit data empty interrupt (TXI) requests, transfer error interrupt (ERI) requests, and receive data full interrupt (RXI) requests. The transmit end interrupt (TEI) request is not used in this mode.

When the TEND flag in SSR is set to 1, a TXI interrupt request is generated.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated.

When any of flags ORER, PER, and ERS in SSR is set to 1, an ERI interrupt request is generated. The relationship between the operating states and interrupt sources is shown in table 15-8.

**Table 15-8 Smart Card Mode Operating States and Interrupt Sources**

| Operating State | | Flag | Enable Bit | Interrupt Source | DMAC Activation | DTC Activation |
|---|---|---|---|---|---|---|
| Transmit Mode | Normal operation | TEND | TIE | TXI | Possible | Possible |
| | Error | ERS | RIE | ERI | Not possible | Not possible |
| Receive Mode | Normal operation | RDRF | RIE | RXI | Possible | Possible |
| | Error | PER, ORER | RIE | ERI | Not possible | Not possible |

**Data Transfer Operation by DMAC or DTC:** In smart card mode, as with the normal SCI, transfer can be carried out using the DMAC or DTC. In a transmit operation, the TDRE flag is also set to 1 at the same time as the TEND flag in SSR, and a TXI interrupt is generated. If the TXI request is designated beforehand as a DMAC or DTC activation source, the DMAC or DTC will be activated by the TXI request, and transfer of the transmit data will be carried out. The TDRE and TEND flags are automatically cleared to 0 when data transfer is performed by the DMAC or DTC. In the event of an error, the SCI retransmits the same data automatically. The TEND flag remains cleared to 0 during this time, and the DMAC is not activated. Thus, the number of bytes specified by the SCI and DMAC are transmitted automatically even in retransmission following an error. However, the ERS flag is not cleared automatically when an error occurs, and so the RIE bit should be set to 1 beforehand so that an ERI request will be generated in the event of an error, and the ERS flag will be cleared.

When performing transfer using the DMAC or DTC, it is essential to set and enable the DMAC or DTC before carrying out SCI setting. For details of the DMAC and DTC setting procedures, see section 7, DMA Controller (DMAC), and section 8, Data Transfer Controller (DTC).

In a receive operation, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. If the RXI request is designated beforehand as a DMAC or DTC activation source, the DMAC or DTC will be activated by the RXI request, and transfer of the receive data will be carried out. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. If an error occurs, an error flag is set but the RDRF flag is not. For this reason, the DMAC

or DTC is not activated, but instead, an ERI interrupt request is sent to the CPU. Therefore, the error flag should be cleared.

### 15.3.7 Example of Use of Software Standby Mode

When using software standby mode in a system that uses smart card interface mode, the following procedure should be followed to maintain the serial clock pulse width.

Figure 15-10 shows an example of the use of software standby mode.

**(1) Transition to software standby mode**

[1] Set DR and DDR of the I/O port corresponding to the serial clock to the value for the fixed output state in software standby mode.

[2] Write 0 to the TE bit and RE bit in SCR to halt transmit/receive operation. At the same time, set the CKE1 bit to the value for the fixed output state in software standby mode.

[3] Write 0 to the CKE0 bit in SCR to halt the clock.

[4] Wait for one serial clock period. During this interval, serial clock output is fixed at the specified level, with the pulse width maintained.

[5] Write H'00 to SMR and SCMR.

[6] Make the transition to software standby mode.

**(2) Exiting software standby mode**

[7] Exit software standby mode by means of an external interrupt.

[8] Set the CKE1 bit in SCR to the value for the fixed output state (corresponding I/O port state) in software standby mode.

[9] Set smart card interface mode and output the clock. The clock is output with the specified pulse width.

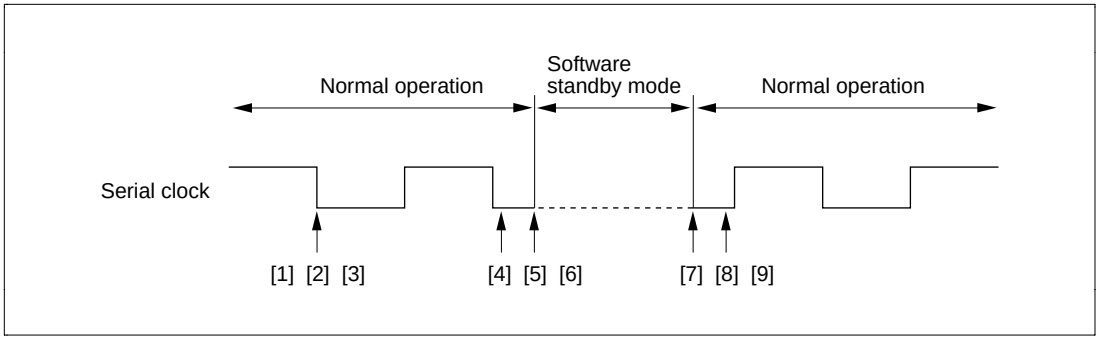



**Figure 15-10 Entering and Exiting Software Standby Mode**

### 15.3.8 Powering On

The following procedure should be used to secure the serial clock pulse width after powering on.

[1] The initial state of the serial clock after powering on is port input and high impedance. Use a pull-up resistor or pull-down resistor to fix the potential.

[2] Specify the output state with the CKE1 bit in SCR.

[3] Set SMR and SCMR, and switch to smart card interface mode.

[4] Set the CKE0 bit in SCR to 1 to start the serial clock output.

## 15.4 Usage Notes

The following points should be noted when using the SCI as a Smart Card interface.

**Receive Data Sampling Timing and Reception Margin in Smart Card Interface Mode:** In Smart Card Interface mode, the SCI operates on a basic clock with a frequency of 372 times the transfer rate.

In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 186th pulse of the basic clock. This is illustrated in figure 15-11.

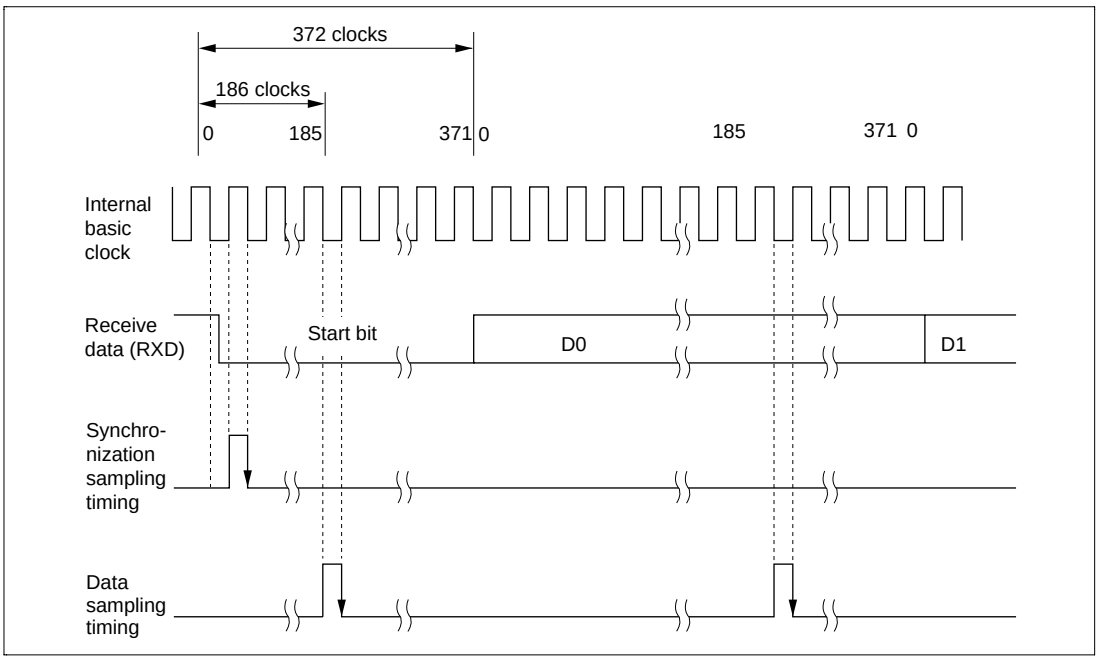



**Figure 15-11 Receive Data Sampling Timing in Smart Card Mode**

Thus the reception margin in asynchronous mode is given by the following formula.

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}\,(1 + F) \right| \times 100\%$$

Where M: Reception margin (%)
N: Ratio of bit rate to clock (N = 372)
D: Clock duty (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in the above formula, the reception margin formula is as follows.

When D = 0.5 and F = 0,

$$M = (0.5 - 1/2 \times 372) \times 100\%$$
$$= 49.866\%$$

**Retransfer Operations:** Retransfer operations are performed by the SCI in receive mode and transmit mode as described below.

- Retransfer operation when SCI is in receive mode

  Figure 15-12 illustrates the retransfer operation when the SCI is in receive mode.

[1] If an error is found when the received parity bit is checked, the PER bit in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The PER bit in SSR should be kept cleared to 0 until the next parity bit is sampled.

[2] The RDRF bit in SSR is not set for a frame in which an error has occurred.

[3] If no error is found when the received parity bit is checked, the PER bit in SSR is not set to 1.

[4] If no error is found when the received parity bit is checked, the receive operation is judged to have been completed normally, and the RDRF flag in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an RXI interrupt request is generated.

  If DMAC or DTC data transfer by an RXI source is enabled, the contents of RDR can be read automatically. When the RDR data is read by the DMAC or DTC, the RDRF flag is automatically cleared to 0.

[5] When a normal frame is received, the pin retains the high-impedance state at the timing for error signal transmission.

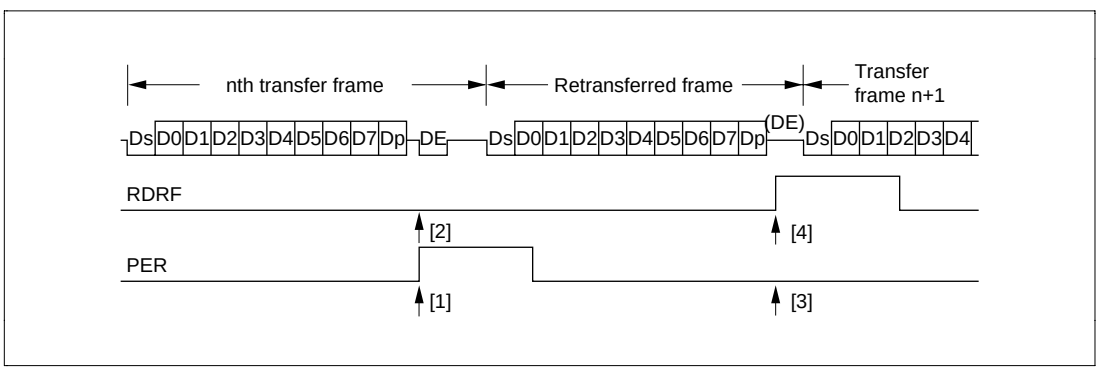



**Figure 15-12 Retransfer Operation in SCI Receive Mode**

- Retransfer operation when SCI is in transmit mode

  Figure 15-13 illustrates the retransfer operation when the SCI is in transmit mode.

[6] If an error signal is sent back from the receiving end after transmission of one frame is completed, the ERS bit in SSR is set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The ERS bit in SSR should be kept cleared to 0 until the next parity bit is sampled.

[7] The TEND bit in SSR is not set for a frame for which an error signal indicating an abnormality is received.

[8] If an error signal is not sent back from the receiving end, the ERS bit in SSR is not set.

[9] If an error signal is not sent back from the receiving end, transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SSR is set to 1. If the TIE bit in SCR is enabled at this time, a TXI interrupt request is generated.

If data transfer by the DMAC or DTC by means of the TXI source is enabled, the next data can be written to TDR automatically. When data is written to TDR by the DMAC or DTC, the TDRE bit is automatically cleared to 0.

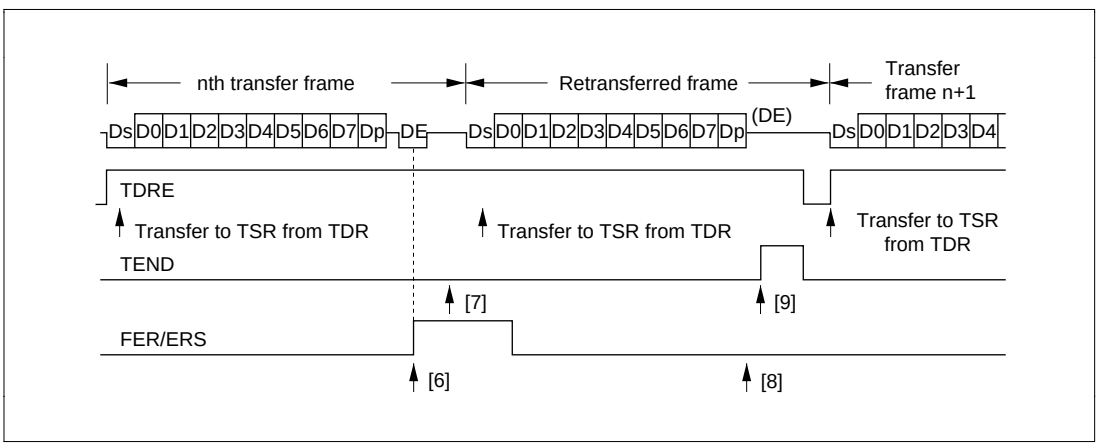



**Figure 15-13 Retransfer Operation in SCI Transmit Mode**

# Section 16 A/D Converter

## 16.1 Overview

This A/D converter has 10-bit resolution, and allows up to 8 analog input channels to be selected.

### 16.1.1 Features

A/D converter features are listed below.

- 10-bit resolution
- Eight input channels
- Settable analog conversion voltage range
  - — Conversion of analog voltages from 0 V to $V_{ref}$, with the reference voltage pin ($V_{ref}$) as the analog reference voltage
- High-speed conversion
  - — Minimum conversion time: 2.2 µs per channel (at 20 MHz operation)
    1.0 µs per channel in continuous conversion
- Variety of conversion modes
  - — Choice of select mode or group mode
  - — Choice of single mode or scan mode
  - — Buffer operation possible
  - — Simultaneous 2-channel sampling possible
- Three kinds of conversion start
  - — Choice of software or timer conversion start trigger (TPU or 8-bit timer), or $\overline{\text{ADTRG}}$ pin
- Eight data registers
  - — Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- A/D conversion end interrupt generation
  - — A/D conversion end interrupt (ADI) request can be generated at the end of A/D conversion

### 16.1.2 Block Diagram

Figure 16-1 shows a block diagram of the A/D converter.

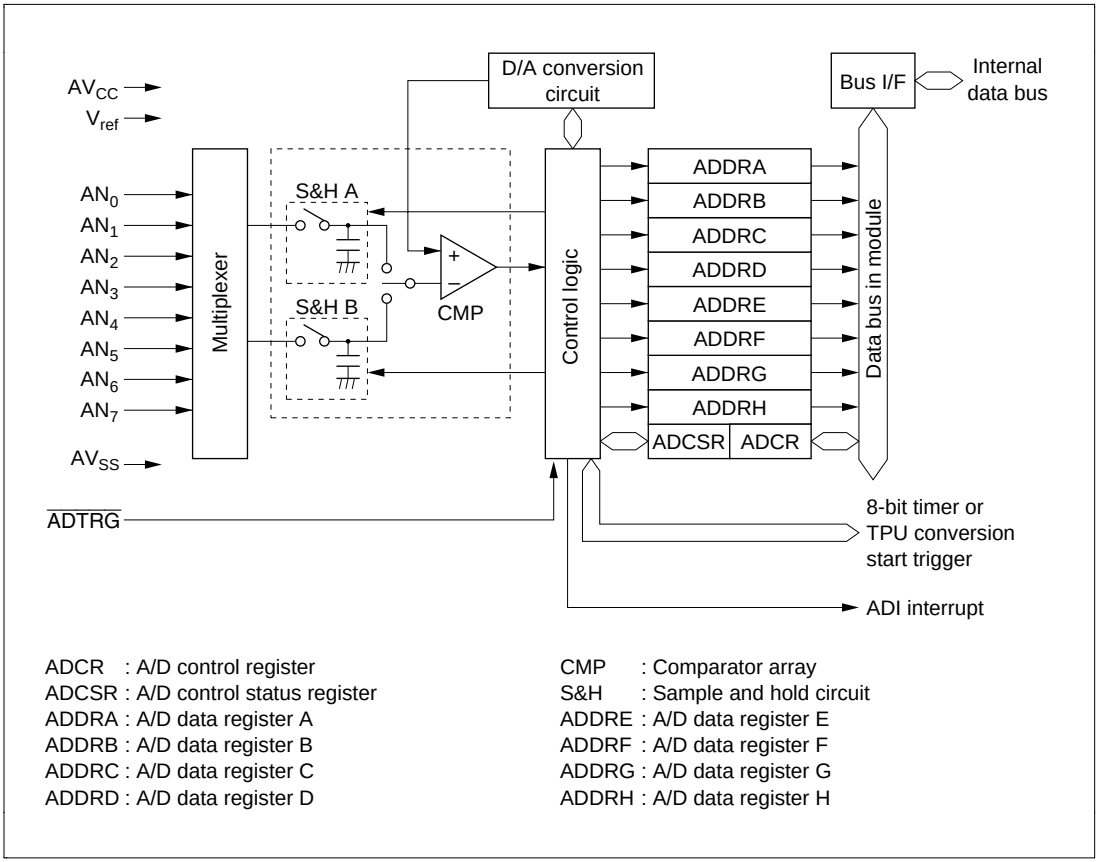



**Figure 16-1 Block Diagram of A/D Converter**

### 16.1.3 Pin Configuration

Table 16-1 summarizes the input pins used by the A/D converter.

The $AV_{CC}$ and $AV_{SS}$ pins are the power supply pins for the analog block in the A/D converter. The $V_{ref}$ pin is the A/D conversion reference voltage pin.

**Table 16-1 A/D Converter Pins**

| Pin Name | Symbol | I/O | Function |
|---|---|---|---|
| Analog power supply | $AV_{CC}$ | Input | Analog block power supply |
| Analog ground | $AV_{SS}$ | Input | Analog block ground and A/D conversion reference voltage |
| Reference voltage | $V_{ref}$ | Input | A/D conversion reference voltage |
| Analog input 0 | $AN_0$ | Input | Analog input channel 0 |
| Analog input 1 | $AN_1$ | Input | Analog input channel 1 |
| Analog input 2 | $AN_2$ | Input | Analog input channel 2 |
| Analog input 3 | $AN_3$ | Input | Analog input channel 3 |
| Analog input 4 | $AN_4$ | Input | Analog input channel 4 |
| Analog input 5 | $AN_5$ | Input | Analog input channel 5 |
| Analog input 6 | $AN_6$ | Input | Analog input channel 6 |
| Analog input 7 | $AN_7$ | Input | Analog input channel 7 |
| A/D external trigger input | $\overline{ADTRG}$ | Input | External trigger for starting A/D conversion |

### 16.1.4 Register Configuration

Table 16-2 summarizes the registers of the A/D converter.

**Table 16-2 A/D Converter Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| A/D data register A | ADDRA | R | H'0000 | H'FF90 |
| A/D data register B | ADDRB | R | H'0000 | H'FF92 |
| A/D data register C | ADDRC | R | H'0000 | H'FF94 |
| A/D data register D | ADDRD | R | H'0000 | H'FF96 |
| A/D data register E | ADDRE | R | H'0000 | H'FF98 |
| A/D data register F | ADDRF | R | H'0000 | H'FF9A |
| A/D data register G | ADDRG | R | H'0000 | H'FF9C |
| A/D data register H | ADDRH | R | H'0000 | H'FF9E |
| A/D control/status register | ADCSR | R/(W)*[2] | H'00 | H'FFA0 |
| A/D control register | ADCR | R/W | H'00 | H'FFA1 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Bit 7 can only be written with 0 for flag clearing.

## 16.2 Register Descriptions

### 16.2.1 A/D Data Registers A to H (ADDRA to ADDRH)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | — | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | — | — | R | R | R | R | R | R | R | R | R | R |

There are eight 16-bit read-only ADDR registers, ADDRA to ADDRH, used to store the results of A/D conversion.

The 10-bit data resulting from A/D conversion is transferred to the ADDR register for the channel on which conversion was performed, and stored there. The lower 8 bits of the converted data are transferred to the lower byte (bits 7 to 0) of ADDR, and the upper 2 bits are transferred to the upper byte (bits 9 and 8). Bits 15 to 10 are always read as 0.

Byte or word length can be selected for data reads. In a byte data read, the upper 8 bits of the converted data are transferred. Buffer operation is also possible by using ADDRA to ADDRD in combination.

The correspondence between the analog input channels and ADDR registers is shown in table 16-3.

The ADDR registers are initialized to H'0000 by a reset, and in hardware standby mode.

**Table 16-3 Analog Input Channels and Corresponding ADDR Registers**

| Analog Input Channel | A/D Data Register |
|---|---|
| $AN_0$ | ADDRA* |
| $AN_1$ | ADDRB* |
| $AN_2$ | ADDRC* |
| $AN_3$ | ADDRD* |
| $AN_4$ | ADDRE |
| $AN_5$ | ADDRF |
| $AN_6$ | ADDRG |
| $AN_7$ | ADDRH |

Note: * Except when buffer operation is used.

### 16.2.2 A/D Control/Status Register (ADCSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ADF | ADIE | ADST | CKS | GRP | CH2 | CH1 | CH0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to bit 7, to clear this flag.

ADCSR is an 8-bit readable/writable register that controls A/D conversion operations and shows the status of the operation.

ADCSR is initialized to H'00 by a reset, and in hardware standby mode.

**Bit 7—A/D End Flag (ADF):** Status flag that indicates the end of A/D conversion.

**Bit 7**

| ADF | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When 0 is written to the ADF flag after reading ADF = 1<br>• When the DTC or DMAC is activated by an ADI interrupt and the prescribed register is read |
| 1 | [Setting conditions]<br>• Single mode: When conversion ends for all specified channels, and A/D conversion ends*<br>• Scan mode: When one round of conversion has been performed on all specified channels |

Note: * In buffer operation, the ADF flag is not set until completion of the specified buffer operation.

**Bit 6—A/D Interrupt Enable (ADIE):** Selects enabling or disabling of interrupt (ADI) requests at the end of A/D conversion.

Only set the ADIE bit while conversion is stopped.

**Bit 6**

| ADIE | Description | |
|---|---|---|
| 0 | A/D conversion end interrupt (ADI) request disabled | (Initial value) |
| 1 | A/D conversion end interrupt (ADI) request enabled | |

**Bit 5—A/D Start (ADST):** Selects starting or stopping on A/D conversion. Holds a value of 1 during A/D conversion.

The ADST bit can be set to 1 by software, a timer conversion start trigger, or the A/D external trigger input pin ($\overline{\text{ADTRG}}$).

**Bit 5**

| ADST | Description | |
|---|---|---|
| 0 | A/D conversion stopped | (Initial value) |
| 1 | • Single mode: A/D conversion is started. Cleared to 0 automatically when conversion on the specified channel ends<br>• Scan mode: A/D conversion is started. Conversion continues until ADST is cleared to 0 by software | |

**Bit 4—Clock Select (CKS):** Sets the A/D conversion time. Set the CKS bit according to the operating frequency so that the conversion time is at least 2 µs.

Only change the conversion time while conversion is stopped.

**Bit 4**

| CKS | Description | |
|---|---|---|
| 0 | Conversion time = 24 states (A/D converter reference clock = ø) | (Initial value) |
| 1 | Conversion time = 44 states (A/D converter reference clock = ø/2) | |

**Bit 3—Group Mode (GRP):** Selects select mode or group mode for A/D conversion channel operation.

Only set the GRP bit while conversion is stopped.

**Bit 3**

| GRP | Description | |
|---|---|---|
| 0 | Select mode | (Initial value) |
| 1 | Group mode | |

**Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0):** Together with the GRP bit, these bits select the analog input channel(s).

Only set the input channel while conversion is stopped.

| Bit 2 | Bit 1 | Bit 0 | Description | |
|---|---|---|---|---|
| **CH2** | **CH1** | **CH0** | **Select Mode (GRP = 0)** | **Group Mode (GRP = 1)** |
| 0 | 0 | 0 | $AN_0$ (Initial value) | $AN_0$ |
| | | 1 | $AN_1$ | $AN_0$ to $AN_1$ |
| | 1 | 0 | $AN_2$ | $AN_0$ to $AN_2$ |
| | | 1 | $AN_3$ | $AN_0$ to $AN_3$ |
| 1 | 0 | 0 | $AN_4$ | $AN_0$ to $AN_4$ |
| | | 1 | $AN_5$ | $AN_0$ to $AN_5$ |
| | 1 | 0 | $AN_6$ | $AN_0$ to $AN_6$ |
| | | 1 | $AN_7$ | $AN_0$ to $AN_7$ |

### 16.2.3 A/D Control Register (ADCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | PWR | TRGS1 | TRGS0 | SCAN | DSMP | BUFE1 | BUFE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ADCR is an 8-bit readable/writable register that controls A/D conversion operations.

ADCR is initialized to H'00 by a reset, and in hardware standby mode.

**Bit 7—Reserved:** Always read as 0. When written to, 0 must be written.

**Bit 6—Power Supply Bit (PWR):** Specifies the A/D converter's conversion start mode. Setting the PWR bit to 1 sets high-speed start mode, while clearing the bit to 0 sets low-power conversion mode. For details of conversion start operations, see section 16.4.7, Conversion Start Modes.

Only set the PWR bit while conversion is stopped.

| Bit 6 | | |
|---|---|---|
| **PWR** | **Description** | |
| 0 | Low-power conversion mode | (Initial value) |
| 1 | High-speed start mode | |

**Bits 5 and 4—Timer Trigger Select 1 and 0 (TRGS1, TRGS0):** These bits select enabling or disabling of A/D conversion start by a trigger signal.

Only set bits TRGS1 and TRGS0 while conversion is stopped.

| Bit 5 | Bit 4 | | |
|---|---|---|---|
| **TRGS1** | **TRGS0** | **Description** | |
| 0 | 0 | A/D conversion start by software is enabled | (Initial value) |
| | 1 | A/D conversion start by TPU conversion start trigger is enabled | |
| 1 | 0 | A/D conversion start by 8-bit timer conversion start trigger is enabled | |
| | 1 | A/D conversion start by external trigger pin ($\overline{\text{ADTRG}}$) is enabled | |

**Bit 3—Scan Mode (SCAN):** Selects single mode or scan mode as the A/D conversion operating mode. For operation in single mode and scan mode, see section 16.4, Operation.

Only set the SCAN bit while conversion is stopped.

| Bit 3 | | |
|---|---|---|
| **SCAN** | **Description** | |
| 0 | Single mode | (Initial value) |
| 1 | Scan mode | |

**Bit 2—Simultaneous Sampling (DSMP):** Enables or disables simultaneous sampling of two channels. For details of simultaneous sampling, see section 16.4.6, Simultaneous Sampling Operation.

Only set the DSMP bit while conversion is stopped.

| Bit 2 | |
|---|---|
| **DSMP** | **Description** |
| 0 | Normal sampling operation (Initial value) |
| 1 | Simultaneous sampling operation |

**Bits 1 and 0—Buffer Enable 1 and 0 (BUFE1, BUFE0):** These bits specify whether or not registers ADDRB to ADDRD are to be used as buffer registers.

For setting and clearing of the ADF flag in the case of buffer operation, see section 16.4.5, Buffer Operation.

Only set the BUFE1 and BUFE0 bits while conversion is stopped.

| Bit 1 | Bit 0 | |
|---|---|---|
| **BUFE1** | **BUFE0** | **Description** |
| 0 | 0 | Normal operation (Initial value) |
| | 1 | ADDRA and ADDRB are used for buffer operation (conversion result → ADDRA → ADDRB)<br>(ADDRB is the buffer register) |
| 1 | 0 | ADDRA and ADDRC, and ADDRB and ADDRD, are used for buffer operation (conversion result 1 → ADDRA → ADDRC; conversion result 2 → ADDRB → ADDRD)<br>(ADDRC and ADDRD are the buffer registers) |
| | 1 | ADDRA to ADDRD are used for buffer operation<br>(conversion result → ADDRA → ADDRB → ADDRC → ADDRD)<br>(ADDRB to ADDRD are the buffer registers) |

### 16.2.4 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP9 bit in MSTPCR is set to 1, A/D converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 9—Module Stop (MSTP9):** Specifies the A/D converter module stop mode.

**Bit 9**

| MSTP9 | Description | |
|---|---|---|
| 0 | A/D converter module stop mode cleared | |
| 1 | A/D converter module stop mode set | (Initial value) |

## 16.3 Interface to Bus Master

ADDRA to ADDRH are 16-bit registers, and the data bus to the bus master is 16 bits wide. The bus master can perform either word-size or byte-size reads on ADDRA to ADDRH.

In a word-size read of an ADDR register, all 16 bits of the ADDR contents are transferred to the bus master in one go. In a byte-size read of the upper byte only, the contents of the upper 8 bits (AD9 to AD2) of the transferred data (AD9 to AD0) are transferred to the bus master.

Figure 16-2 illustrates the operation when reading an ADDR register.

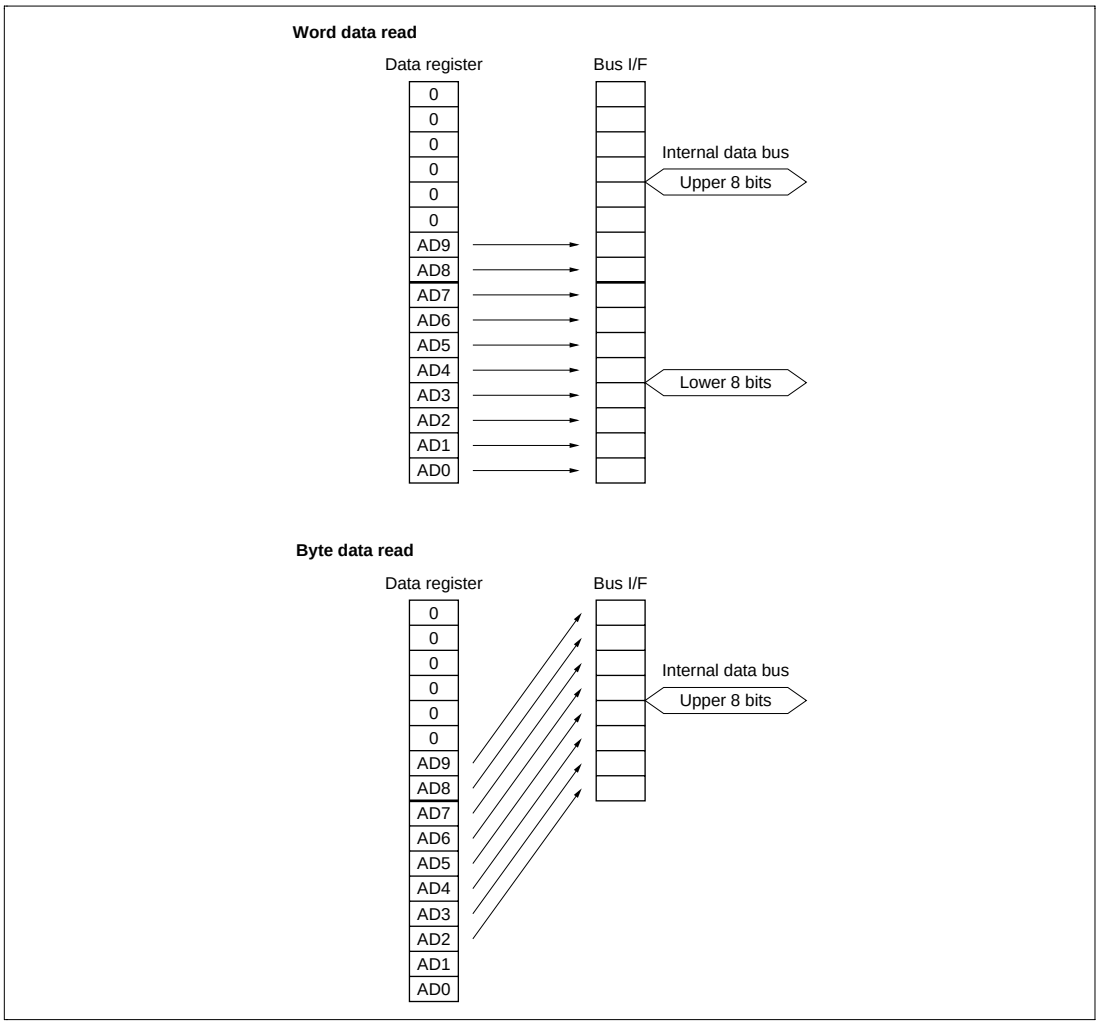



**Figure 16-2 ADDR Read Operation**

## 16.4 Operation

The A/D converter has 10-bit resolution.

There are four operating modes—select or group, and single or scan—which can be combined with buffer operation or simultaneous sampling operation.

A single channel is selected in select mode, and a number of channels in group mode. In select mode, a single activation results in conversion on all the selected channels, while in scan mode, a single activation results in conversion repeated until stopped by software. In buffer operation, when conversion ends for the channel concerned, the previous conversion results are saved in a buffer register. In simultaneous sampling operation, analog input voltages are sampled on two channels simultaneously, and converted sequentially.

A software or timer conversion start trigger (TPU or 8-bit timer), or $\overline{\text{ADTRG}}$ input, can be selected as the conversion start condition.

Either high-speed start mode or low-power conversion mode can be selected for A/D conversion by means of the PWR bit.

The operating mode or input channel can be changed by rewriting ADCSR and ADCR while the ADST bit is cleared to 0. After ADCSR and ADCR have been rewritten, A/D conversion is started again when the ADST bit is set to 1. A change of operating mode or input channel and ADST bit setting can be carried out simultaneously. A/D conversion can be stopped midway by clearing the ADST bit to 0.

### 16.4.1 Select Single Mode

Select single mode is selected when A/D conversion is to be performed on a single channel only.

A/D conversion is started when the ADST bit is set to 1, according to the specified conversion start condition. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends.

On completion of conversion, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by writing 0 after reading ADCSR. (It can be cleared with the BCLR instruction.)

Figure 16-3 shows an example of A/D converter operation when $AN_1$ is selected in select single mode.

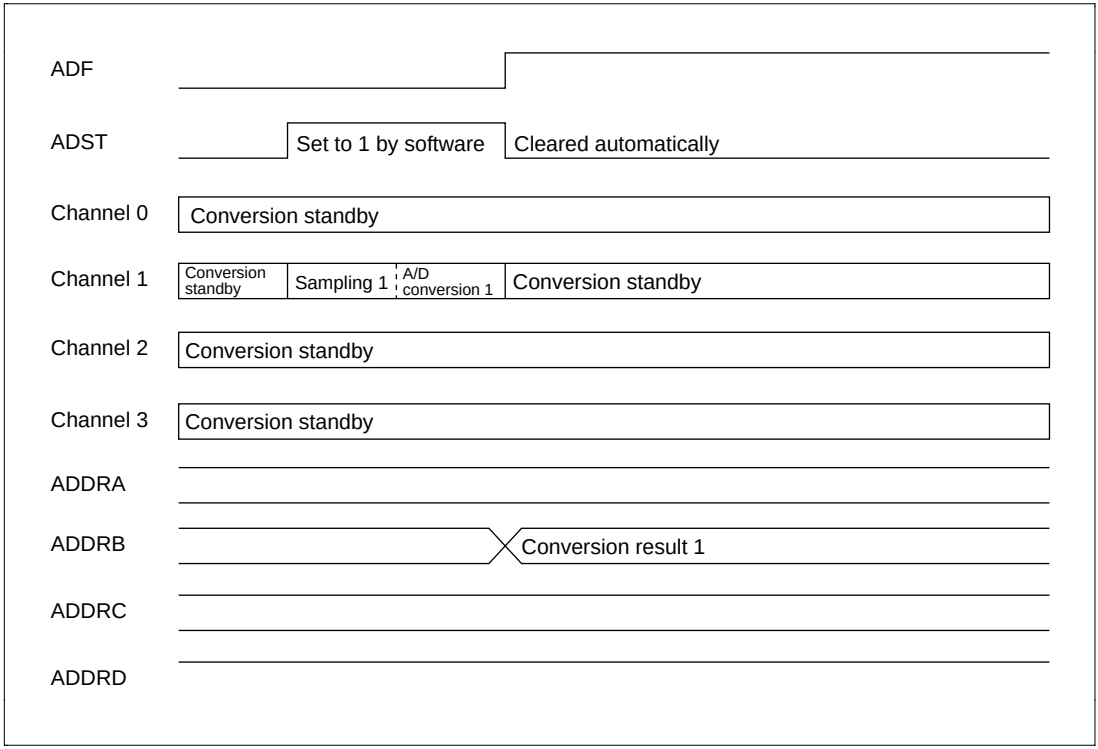



**Figure 16-3 Example of A/D Converter Operation (Select Single Mode)**

### 16.4.2 Select Scan Mode

Select scan mode is selected when A/D conversion is to be performed repeatedly on a single channel. This mode is suitable for constantly monitoring analog input on a single channel.

A/D conversion is started when the ADST bit is set to 1, according to the specified conversion start condition. The ADST bit remains set to 1 until cleared to 0 by software. During this time, A/D conversion is performed repeatedly on the selected input channel.

When the first conversion operation ends, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated and A/D conversion is temporarily halted. Clearing the ADF flag to 0 when conversion has been halted by an ADI interrupt request will restart conversion. The ADF flag is cleared by writing 0 after reading ADCSR. (It can be cleared with the BCLR instruction.)

Figure 16-4 shows an example of A/D converter operation when $AN_1$ is selected in select scan mode.

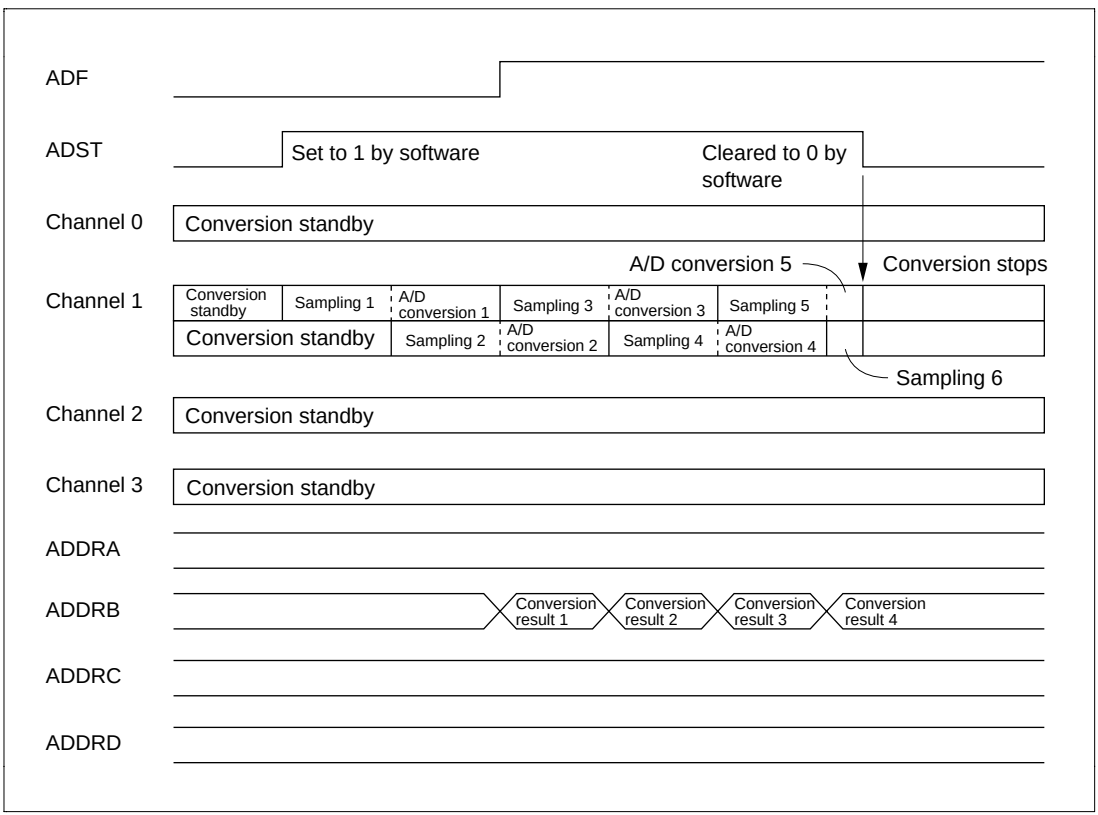



**Figure 16-4 Example of A/D Converter Operation (Select Scan Mode)**

### 16.4.3 Group Single Mode

Group single mode is selected when A/D conversion is to be performed on a number of channels.

A/D conversion is started when the ADST bit is set to 1, according to the specified conversion start condition. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends for all the specified input channels.

On completion of conversion for all the specified input channels the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by writing 0 after reading ADCSR. (It can be cleared with the BCLR instruction.)

Figure 16-5 shows an example of A/D converter operation when $AN_0$ to $AN_2$ are selected in group single mode.

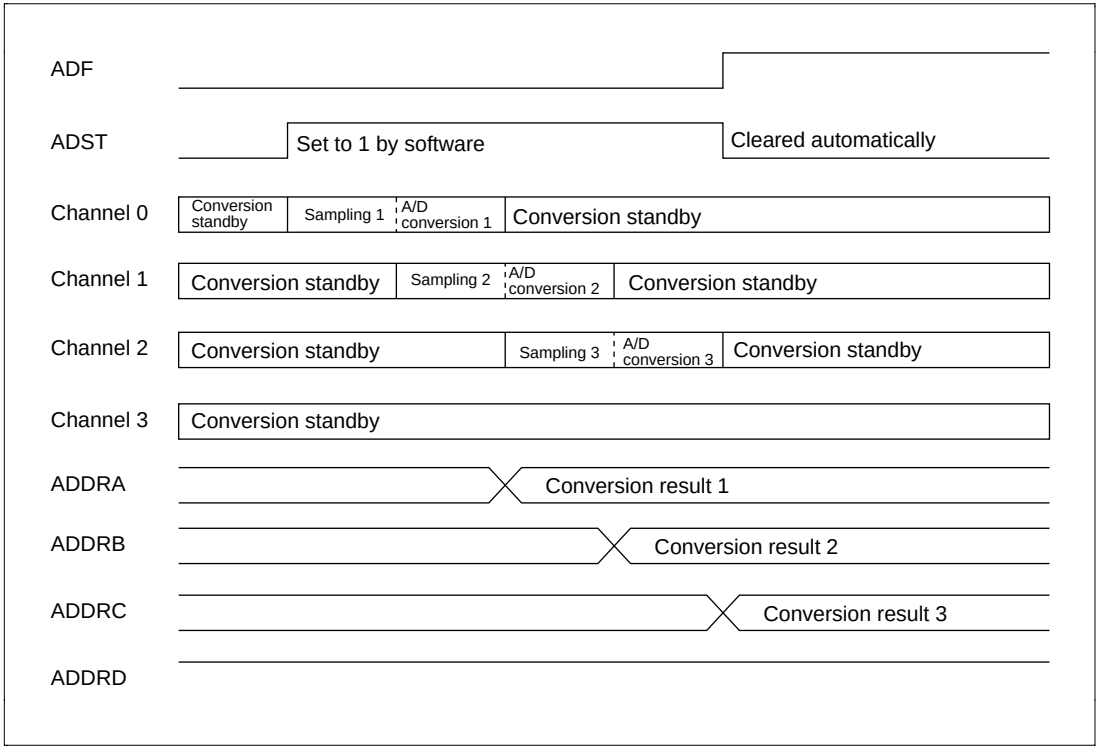



**Figure 16-5 Example of A/D Converter Operation (Group Single Mode)**

### 16.4.4 Group Scan Mode

Group scan mode is selected when A/D conversion is to be performed repeatedly on a number of channels. This mode is suitable for constantly monitoring analog input on a number of channels.

A/D conversion is started when the ADST bit is set to 1, according to the specified conversion start condition. The ADST bit remains set to 1 until cleared to 0 by software. During this time, A/D conversion is performed repeatedly on the selected input channels.

When the first conversion operation ends for all the selected input channels, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated and A/D conversion is temporarily halted. Clearing the ADF flag to 0 when conversion has been halted by an ADI interrupt will restart conversion. The ADF flag is cleared by writing 0 after reading ADCSR. (It can be cleared with the BCLR instruction.)

Figure 16-6 shows an example of A/D converter operation when $AN_0$ to $AN_2$ are selected in group scan mode.

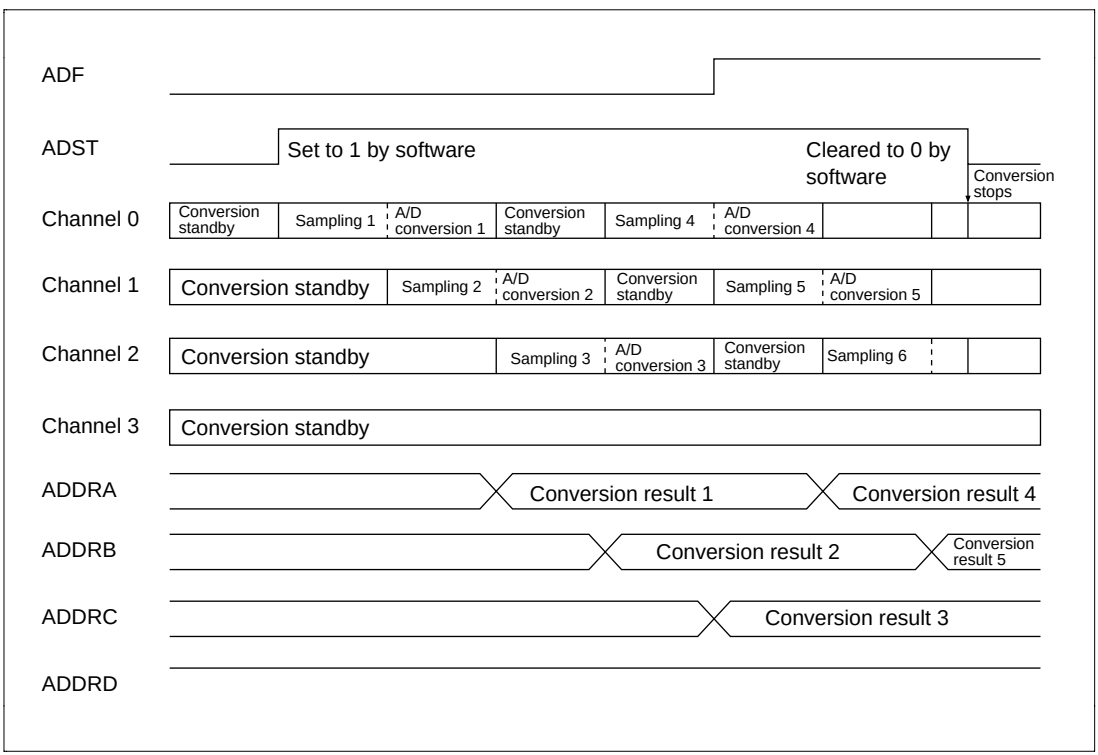



**Figure 16-6 Example of A/D Converter Operation (Group Scan Mode)**

### 16.4.5 Buffer Operation

In buffer operation, when conversion ends on the channel concerned, at the same time as the conversion result is stored in an ADDR register, the previously stored conversion result is transferred to another ADDR register.

There is a choice of three kinds of buffer operation: a two-stage operation, AN0 → ADDRA → ADDRB; dual two-stage operations, AN0 → ADDRA → ADDRC and AN1 → ADDRB → ADDRD; and a four-stage operation, AN0 → ADDRA → ADDRB → ADDRC → ADDRD.

When using buffer operation in combination with simultaneous sampling operation, set GRP=1, BUFE1, BUFE0=B'10, and CH2=0.

Figure 16-7 shows buffer operation timing.

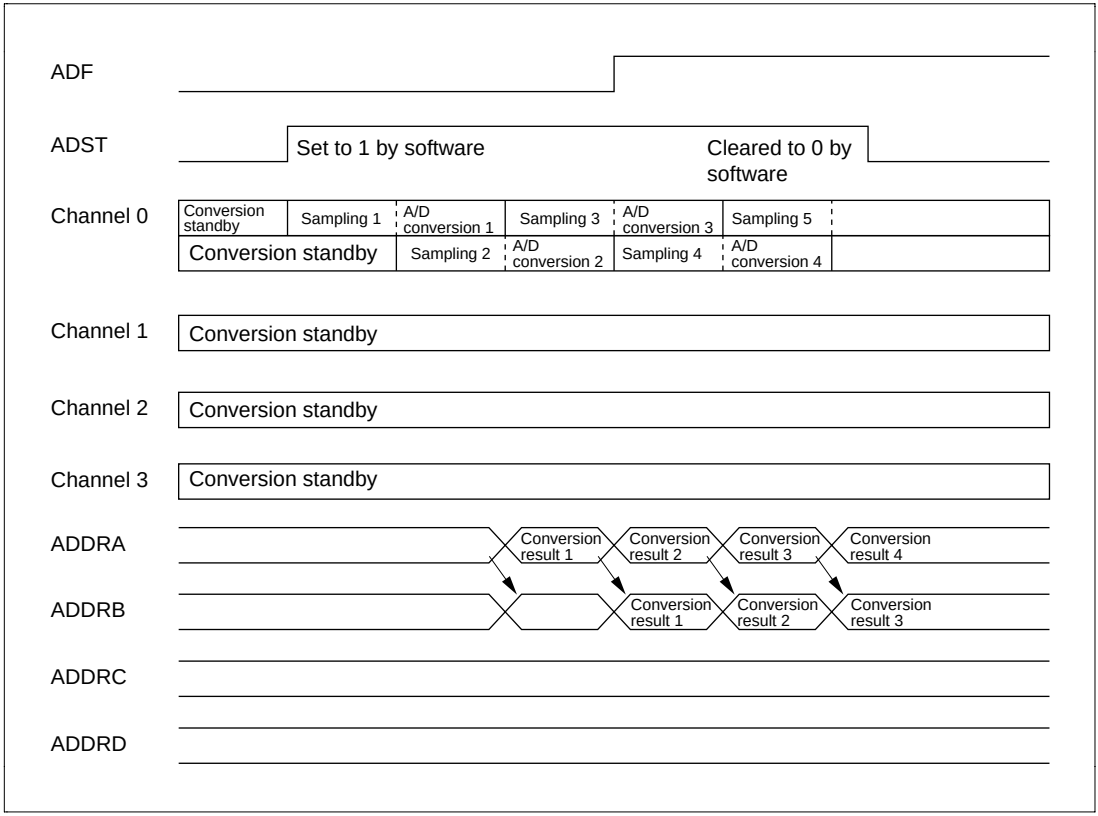



**Figure 16-7 Example of Buffer Operation (Select Scan Mode: Two-Stage Operation, CH2 to CH0 = B'001)**

**Using Buffer Operation Only:** When conversion is performed only on the analog input channels ($AN_0$, $AN_1$) specified by bits BUFE1 and BUFE0, the ADF flag setting condition can be selected by selecting group mode and setting bits CH2 to CH0.

Table 16-4 (1) shows the conversion operation and ADF flag setting conditions in buffer operation. The ADF flag is set on completion of the last conversion shown in the table. In single mode, conversion is halted after the ADF flag is set to 1. In scan mode, conversion continues and the conversion data is stored in order in the buffer register specified by the BUFE1 and BUFE0 bits.

If the ADIE bit is set to 1 while the ADF flag is set to 1, an ADI interrupt is generated. The ADF flag is cleared to 0 by writing 0 after reading ADCSR. (It can be cleared to 0 with the BCLR instruction.)

In select single mode, the conversion wait state is entered on completion of each conversion. When A/D conversion is restarted by software, a timer trigger, or an external trigger, and the number of conversions shown in table 16-4 (1) have been completed, the ADF flag is set to 1.

**Table 16-4 (1) Conversion Channels and ADF Flag Setting/Clearing Conditions in Buffer Operation**

| Setting of CH2 to CH0 | | | Buffer Operation Selection | | |
|---|---|---|---|---|---|
| **CH2** | **CH1** | **CH0** | **BUFE1, 0=B'01** | **BUFE1, 0=B'10** | **BUFE1, 0=B'11** |
| 0 | 0 | 0 | $AN_0$ once (ADDRA) | $AN_0$ and $AN_1$ once each (ADDRB) | $AN_0$ once (ADDRA) |
| | | 1 | $AN_0$ twice (ADDRB) | | $AN_0$ twice (ADDRB) |
| | 1 | 0 | See table 16-4 (2) | AN0 and AN1 twice each (ADDRD) | $AN_0$ three times (ADDRC) |
| | | 1 | | | $AN_0$ four times (ADDRD) |
| 1 | — | — | See table 16-4 (2) | | |

**Combining Group Mode with Buffer Operation:** Bits CH2 to CH0 can be set to perform continuous conversion on the analog input channels ($AN_0$, $AN_1$) specified by bits BUFE1 and BUFE0, and $AN_4$ to $AN_7$.

Table 16-4 (2) shows the conversion operation and ADF flag setting conditions in buffer operation. The ADF flag is set on completion of the last conversion shown in the table. In this case, the analog input corresponding to the ADDR specified in the buffer register is not converted. For example, if BUFE1 and BUFE0 = B'11, and CH2 to CH0 = B'110, the conversion results are stored in ADDRA and ADDRE to ADDRG. The contents of ADDRA to ADDRC prior to the start of conversion are stored in ADDRB to ADDRD.

In single mode, conversion is halted after the ADF flag is set to 1. In scan mode, conversion continues.

**Table 16-4 (2) Conversion Channels and ADF Setting/Clearing Conditions in Buffer Operation**

| Setting of CH2 to CH0 | | | Buffer Operation Selection | | |
|---|---|---|---|---|---|
| **CH2** | **CH1** | **CH0** | **BUFE1, 0 = B'01** | **BUFE1, 0 = B'10** | **BUFE1, 0 = B'11** |
| 0 | 0 | — | See table 16-4 (1) | | |
| | 1 | 0 | $AN_0$, $AN_2$ (ADDRC) | See table 16-4 (1) | |
| | | 1 | $AN_0$, $AN_2$, $AN_3$ (ADDRD) | | |
| 1 | 0 | 0 | $AN_0$, $AN_2$ to $AN_4$ (ADDRE) | $AN_0$, $AN_1$, $AN_4$ (ADDRE) | $AN_0$, $AN_4$ (ADDRE) |
| | | 1 | $AN_0$, $AN_2$ to $AN_5$ (ADDRF) | $AN_0$, $AN_1$, $AN_4$, $AN_5$ (ADDRF) | $AN_0$, $AN_4$, $AN_5$ (ADDRF) |
| | 1 | 0 | $AN_0$, $AN_2$ to $AN_6$ (ADDRG) | $AN_0$, $AN_1$, $AN_4$ to $AN_6$ (ADDRG) | $AN_0$, $AN_4$ to $AN_6$ (ADDRG) |
| | | 1 | $AN_0$, $AN_2$ to $AN_7$ (ADDRH) | $AN_0$, $AN_1$, $AN_4$ to $AN_7$ (ADDRH) | $AN_0$, $AN_4$ to $AN_7$ (ADDRH) |

**Clearing the ADF Flag:** If the DTC or DMAC is activated by an A/D conversion end interrupt, the ADF flag is cleared when the ADDR specified in table 16-4 is read.

**To Reset the Number of Buffer Operations:** Suspend the conversion wait state or conversion, and clear the BUFE1 and BUFE0 bits to B'00. The buffer count will be cleared to 0.

**To Change the Buffer Operation:** Suspend the conversion wait state or conversion, and clear the BUFE1 and BUFE0 bits to B'00. When the BUFE1 and BUFE0 bits are set and conversion is restarted, the buffer operation shown in table 16-4 will be performed.

### 16.4.6 Simultaneous Sampling Operation

In simultaneous sampling operation, the input voltages of two channels are sampled simultaneously, and continuous conversion is performed. Simultaneous sampling operation is enabled in group mode. The channels involved in simultaneous sampling operation are determined by bits CH2 and CH1. The combinations of these bits are shown in table 16-5.

For example, simultaneous sampling will be performed when CH2 and CH1 = B'11, on channel pairs $AN_0$, $AN_1 \rightarrow AN_2$, $AN_3 \rightarrow AN_4$, $AN_5 \rightarrow AN_6$, $AN_7$ in that order if GRP=1. Simultaneous sampling timing is shown in figure 16-8.

**Table 16-5 Simultaneous Sampling Channels**

| Channel Setting | | Sampled Channels |
|---|---|---|
| **CH2** | **CH1** | **GRP = 1** |
| 0 | 0 | $AN_0$, $AN_1$ |
| | 1 | $AN_0$, $AN_1 \rightarrow AN_2$, $AN_3$ |
| 1 | 0 | $AN_0$, $AN_1 \rightarrow AN_2$, $AN_3 \rightarrow AN_4$, $AN_5$ |
| | 1 | $AN_0$, $AN_1 \rightarrow AN_2$, $AN_3 \rightarrow AN_4$, $AN_5 \rightarrow AN_6$, $AN_7$ |

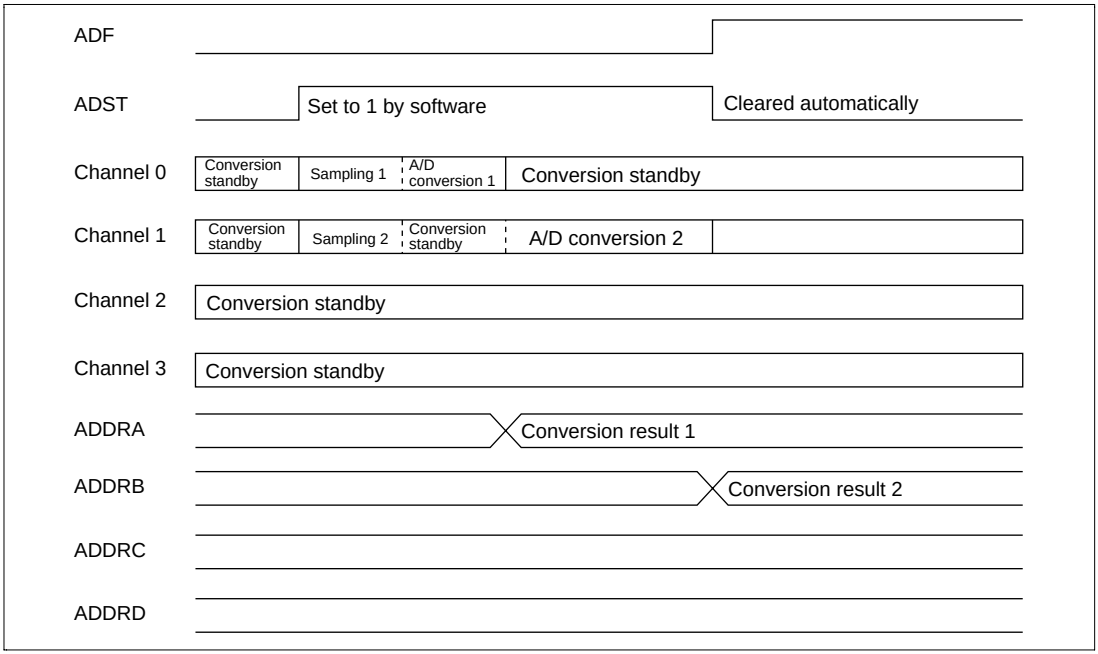



**Figure 16-8 Example of Simultaneous Sampling Operation (Group Single Mode)**

### 16.4.7 Conversion Start Modes

The A/D converter's conversion start mode is set by means of the PWR bit in ADCSR. When the PWR bit is cleared to 0, low-power conversion mode is set, and the internal analog circuitry is made inactive. When the PWR bit is set to 1, high-speed start mode is set, and the analog circuitry is made active.

In low-power conversion mode, the analog circuit power is turned on simultaneously with the start of conversion (ADST setting), and after 200 cycles of the reference clock the analog circuitry changes to the ready state, and the first A/D conversion operation is started. The reference clock is selected by the CKS bit in ADCSR. When conversion is carried out continuously, the second and subsequent A/D conversion operations are performed every 10 cycles. When A/D conversion ends, ADST is cleared to 0 and the analog circuit power is cut automatically. Since the analog circuitry is only active during A/D conversion in this mode, current dissipation can be reduced.

In high-speed start mode, even when A/D conversion ends and ADST is cleared to 0, power continues to be supplied to the analog circuitry and conversion can still be carried out. Conversion is started as soon as ADST is set to 1 again. Only in the case of the first conversion after the analog power supply is turned on, conversion does not begin until 200 cycles after ADST is set. The analog circuit power supply is turned off by clearing the PWR bit to 0. When conversion is carried out continuously, the second and subsequent A/D conversion operations are performed every 10 cycles. High-speed A/D conversion can be achieved in this mode since the analog circuitry is always active.

Figures 16-9 and 16-10 show conversion start operation timing.

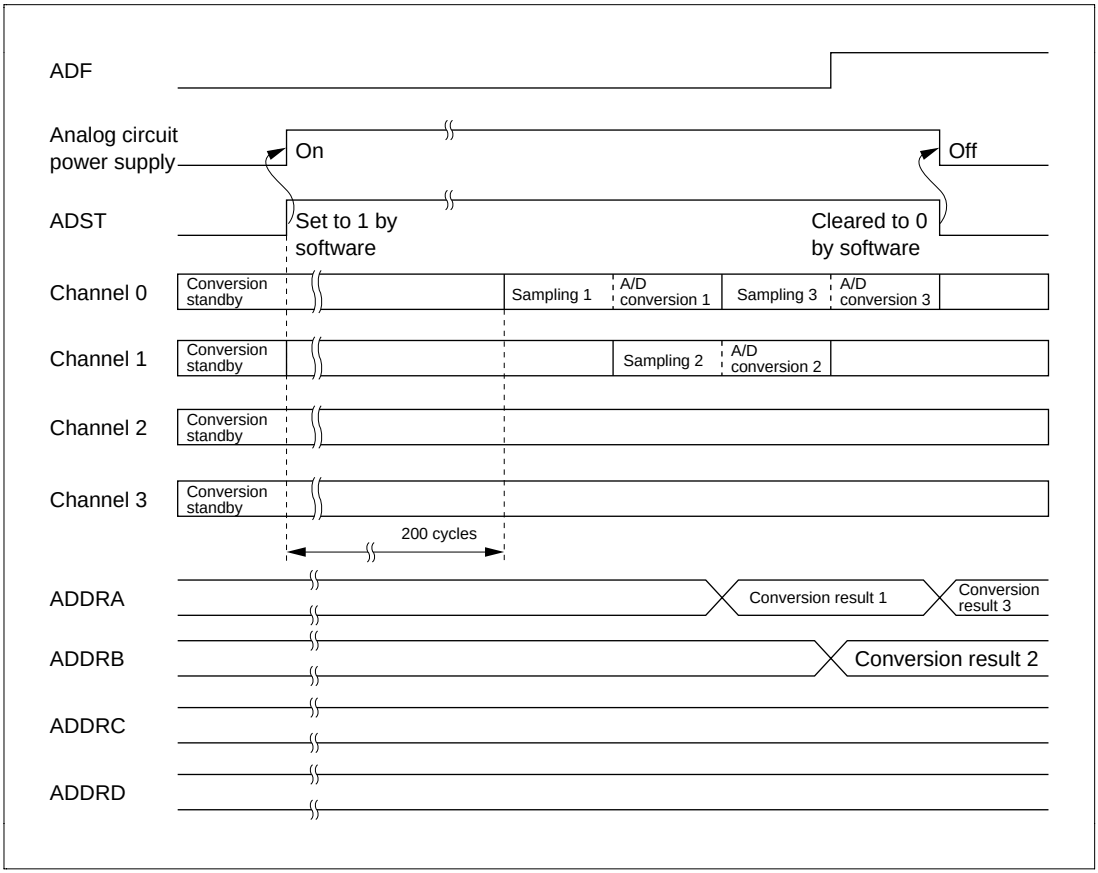



**Figure 16-9 Conversion Start Operation (Low-Power Conversion Mode)**

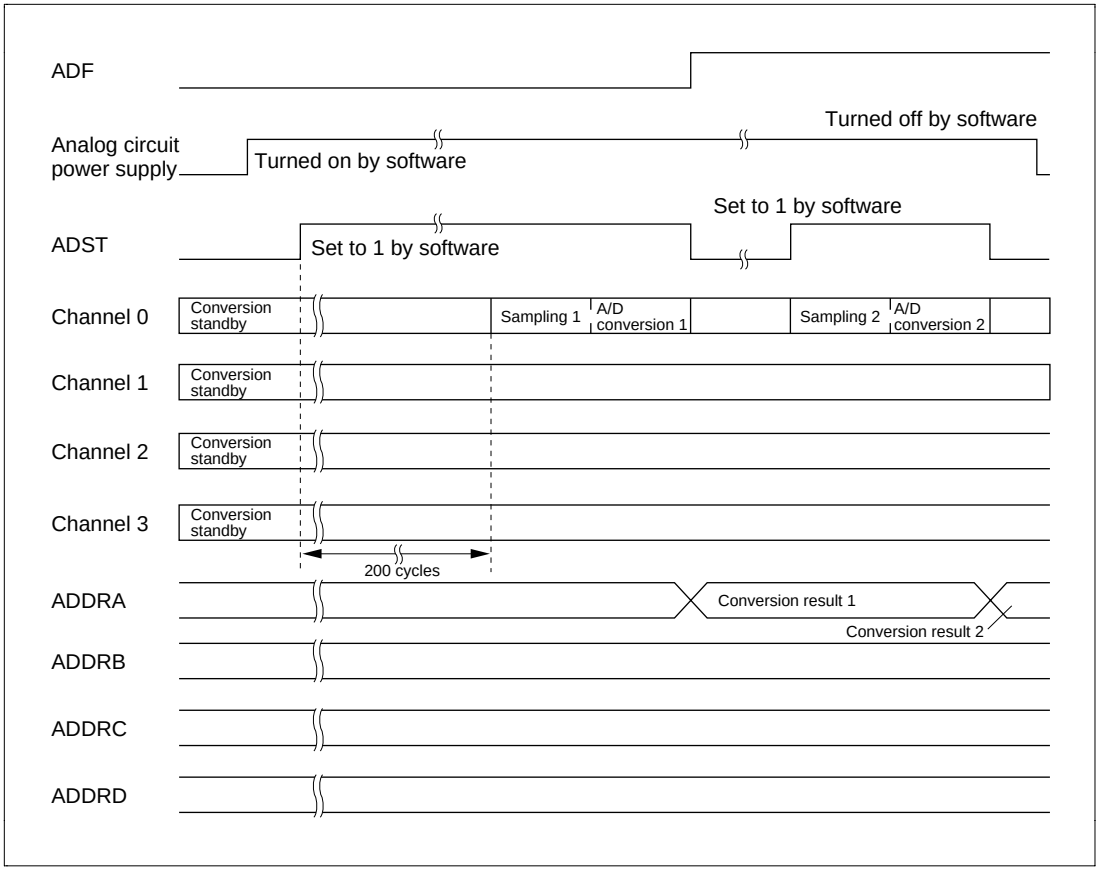



**Figure 16-10 Conversion Start Operation (High-Speed Start Mode)**

### 16.4.8 Starting Conversion by External Input

A/D conversion can be started by a timer conversion start trigger or a trigger signal generated by $\overline{\text{ADTRG}}$ input. When the trigger signal specified by bits TRGS1 and TRGS0 in ADCR is generated, the ADST bit in ADCSR is set to 1 and A/D conversion is started.

Other operations are the same as when the ADST bit is set to 1 by software.

Figure 16-11 shows the timing for setting of the ADST bit by external input.

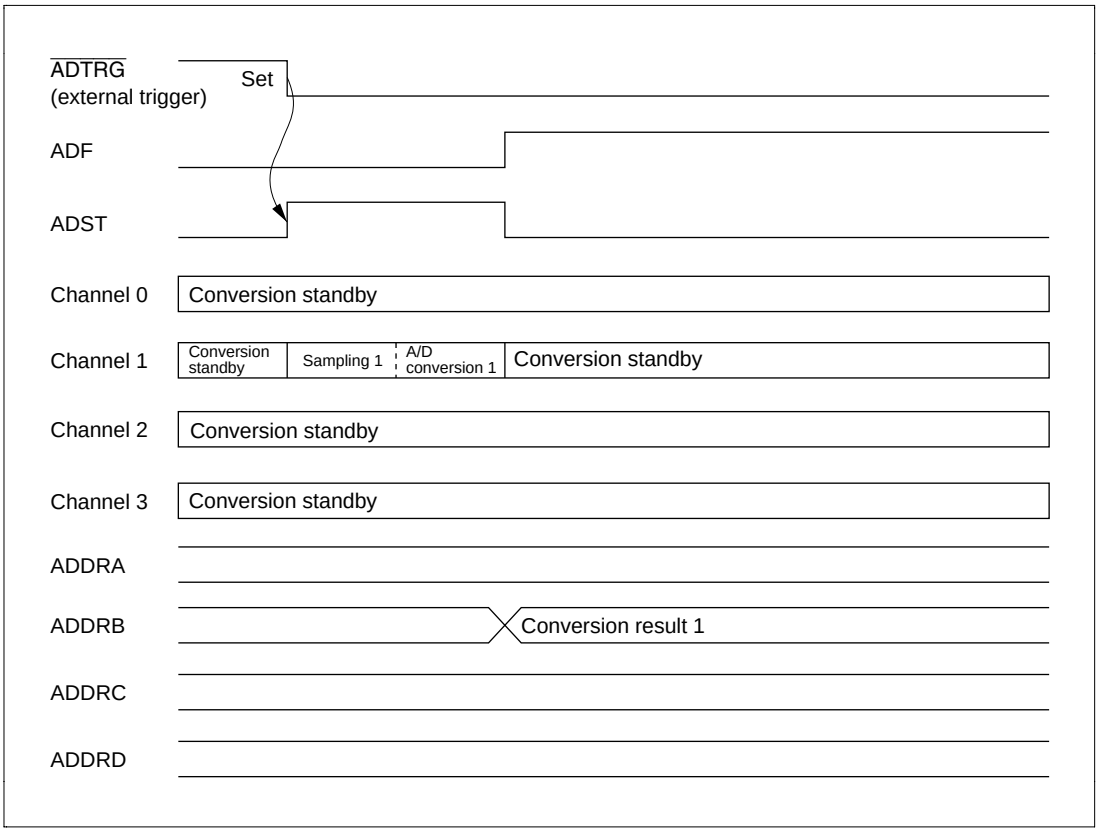



**Figure 16-11 Start of Conversion by $\overline{\text{ADTRG}}$ Conversion Start Trigger**

### 16.4.9 A/D Conversion Time

The A/D converter has a built-in sample and hold circuit. The A/D converter performs input sampling after the elapse of time $t_D$ from the point at which the ADCSR write operation or timer compare match that set the ADST bit to 1, and then starts conversion.

The A/D conversion time $t_{CONV}$ is the sum of the conversion start delay time $t_D$, the input sampling time $t_{SPL}$, the operation time $t_{CP}$, and the ADF flag set delay time $t_F$.

A/D conversion timing is shown in figures 16-12 (1) and (2), and A/D conversion times in table 16-6.

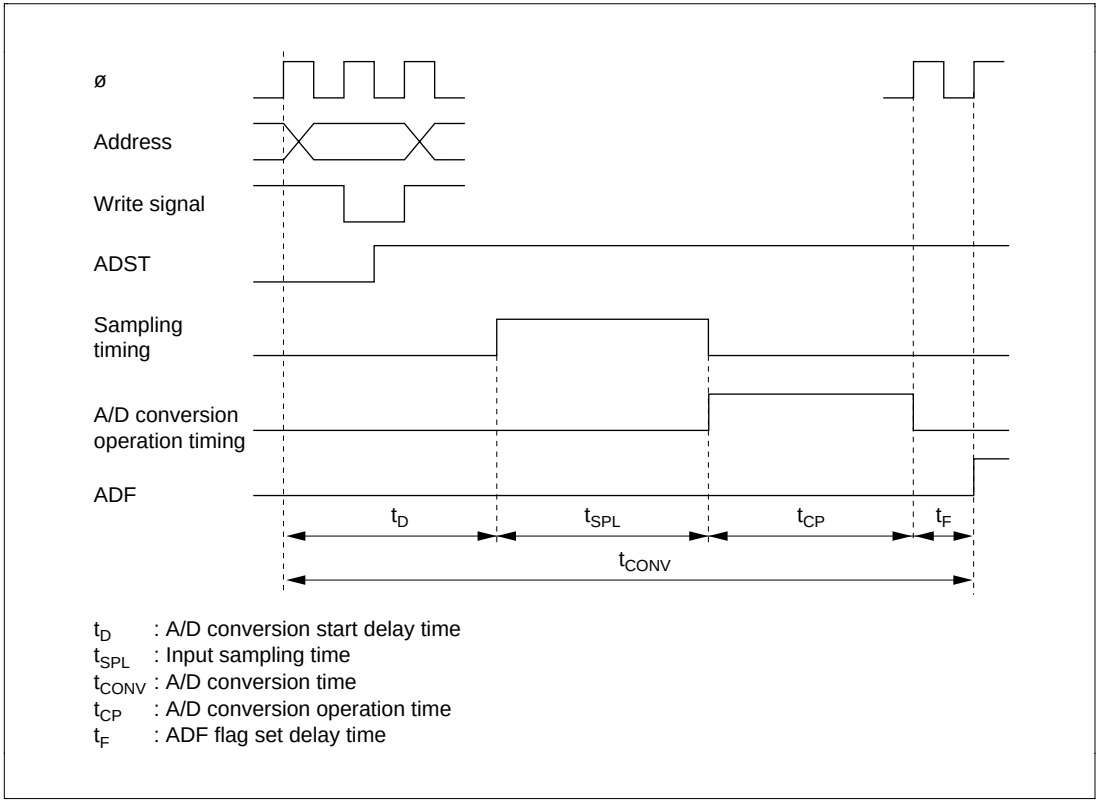
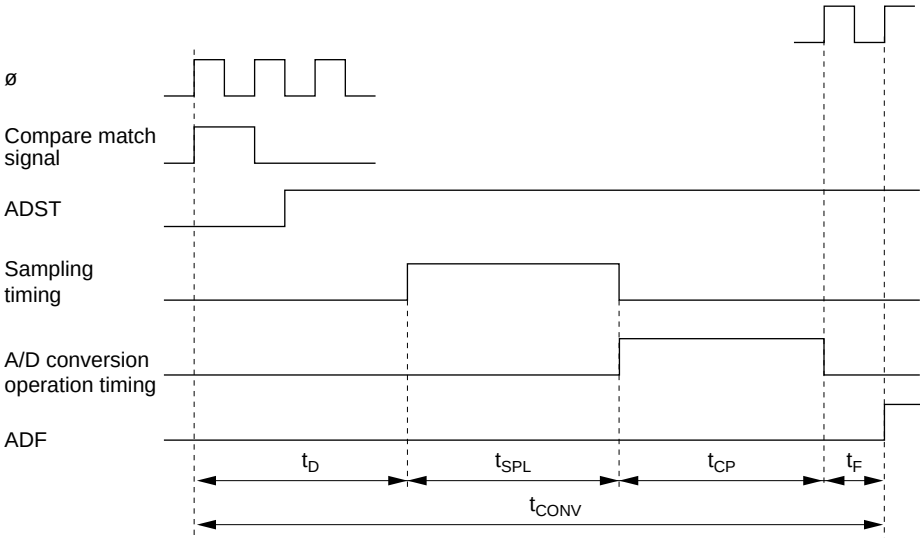



**Figure 16-12 (1) A/D Conversion Timing**




$t_D$ : A/D conversion start delay time
$t_{SPL}$ : Input sampling time
$t_{CONV}$ : A/D conversion time
$t_{CP}$ : A/D conversion operation time
$t_F$ : ADF flag set delay time

**Figure 16-12 (2) A/D Conversion Timing**

**Table 16-6 A/D Conversion Times**

| | Symbol | CKS = 0 | CKS = 1 |
|---|---|---|---|
| A/D conversion start delay time | $t_D$ | 3 | 3 |
| Input sampling time | $t_{SPL}$ | 10 | 20 |
| A/D conversion operation time | $t_{CP}$ | 10 | 20 |
| ADF flag set delay time | $t_F$ | 1 | 1 |
| A/D conversion time | $t_{CONV}$ | 24 | 44 |

Note: Units: States

The figures in the table are for when PWR = 1. If 200 states have not elapsed since setting of the PWR bit, conversion is not performed until 200 states have elapsed. When PWR = 0, 200 states should be added to the first A/D conversion start delay time. When conversion is carried out continuously, the second and subsequent $t_{CONV}$ values are obtained by subtracting $t_{SPL}$.

The CKS bit in ADCSR should be set to give the operation time, $t_{CONV}$. Tables 16-7 (1) and (2) show operating frequencies and CKS bit settings.

(1) When $AV_{CC} \geq 4.5$ V, $t_{CONV} \geq 2$ µs

Condition : $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{ss}$ = AV= 0 V

**Table 16-7 (1) Operating Frequencies and CKS Bit Settings**

| CKS | Conversion Time (States) | Minimum Conversion Time (µs) | | | | |
|---|---|---|---|---|---|---|
| | | 20 MHz | 16 MHz | 10 MHz | 8 MHz | 2 MHz |
| 0 | 24 | — | — | 2.4 | 2.8 | 12.0 |
| 1 | 44 | 2.2 | 2.8 | 4.4 | 5.5 | 22.0 |

Note: — : Cannot be set.

(2) When $AV_{CC} < 4.5$ V, $t_{CONV} \geq 4$ µs

Condition : $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 V to 4.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{ss}$ = $AV_{ss}$ = 0 V

**Table 16-7 (2) Operating Frequencies and CKS Bit Settings**

| CKS | Conversion Time (States) | Minimum Conversion Time (µs) | | | | |
|---|---|---|---|---|---|---|
| | | 10 MHz | 8 MHz | 5 MHz | 4 MHz | 2 MHz |
| 0 | 24 | — | — | 4.8 | 6.0 | 12.0 |
| 1 | 44 | 4.4 | 5.5 | 8.8 | 11.0 | 22.0 |

Note: — : Cannot be set.

## 16.5 Interrupts

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. ADI interrupt requests can be enabled or disabled by means of the ADIE bit in ADCSR.

The DTC or DMAC can be activated by an ADI interrupt. Having the converted data read by the DTC or DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

The A/D converter interrupt source is shown in table 16-8.

If the ADIE bit is set to 1 in scan mode, setting the ADF flag to 1 will temporarily halt A/D conversion. A/D conversion is restarted when the ADF flag is cleared to 0.

When the DTC or DMAC is activated by an ADI interrupt and the last of the specified data registers is read, the ADF flag is cleared to 0.

**Table 16-8 A/D Converter Interrupt Source**

| Interrupt Source | Description | DTC or DMAC Activation |
|---|---|---|
| ADI | Interrupt due to end of conversion | Possible |

## 16.6 Usage Notes

The following points should be noted concerning the A/D converter.

- Analog input voltage range

  The voltage applied to analog input pins $AN_0$ to $AN_7$ during A/D conversion should be in the range $AV_{SS} \leq AN_0$ to $AN_7 \leq AV_{CC}$.

- $AV_{CC}$ and $AV_{SS}$ input voltages

  The $AV_{CC}$ and $AV_{SS}$ input voltages should be set as follows: $AV_{CC} = V_{CC} \pm 10\%$, $AV_{SS} = V_{SS}$. When the A/D converter is not used, set $AV_{CC} = V_{CC}$, $AV_{SS} = V_{SS}$. In standby mode, set $V_{RAM} \leq AV_{CC} \leq 5.5$ V, $AV_{SS} = V_{SS}$ (where $V_{RAM}$ is the RAM standby voltage).

- $V_{ref}$ input voltage

  The analog reference voltage $V_{ref}$ should be set as follows: $V_{ref} \leq AV_{CC}$. When the A/D converter is not used, set $V_{ref} = V_{CC}$. In standby mode, set $V_{RAM} \leq V_{ref} \leq AV_{CC}$ (where $V_{RAM}$ is the RAM standby voltage).

- Input ports

  When a circuit is connected to an input port, the constant should be set to a value less than the A/D converter sampling time. If the constant is large in the case of a circuit, the input voltage may not be sampled properly.

- Conversion start mode

  There is a difference in the current dissipation between high-speed start mode and low-power conversion mode selected for A/D conversion operation according to the PWR bit setting.

- If conversion is terminated while in progress (when ADST = 1) in high-speed start mode (PWR = 1) by clearing the ADST bit, there may be a large degree of error in the first conversion following the restart (on channel 1 in normal mode, or channel 2 in simultaneous sampling mode). If A/D conversion is terminated while in progress, use one of the following methods to solve this problem:

  1. Ignore the first data following a restart after termination.
  2. If the first data after a restart is used, perform a single dummy conversion in single mode after the termination, terminate A/D conversion, then restart.
  3. After termination, write 0 to the PWR bit before restarting. (In this case, the first conversion after the restart will begin after the elapse of 200 A/D reference clock cycles.)

# Section 17 D/A Converter

## 17.1 Overview

The H8S/2655 Series includes a two-channel D/A converter.

### 17.1.1 Features

D/A converter features are listed below

- 8-bit resolution
- Two output channels
- Maximum conversion time of 10 µs (with 20 pF load)
- Output voltage of 0 V to Vref
- D/A output hold function in software standby mode

### 17.1.2 Block Diagram

Figure 17-1 shows a block diagram of the D/A converter.

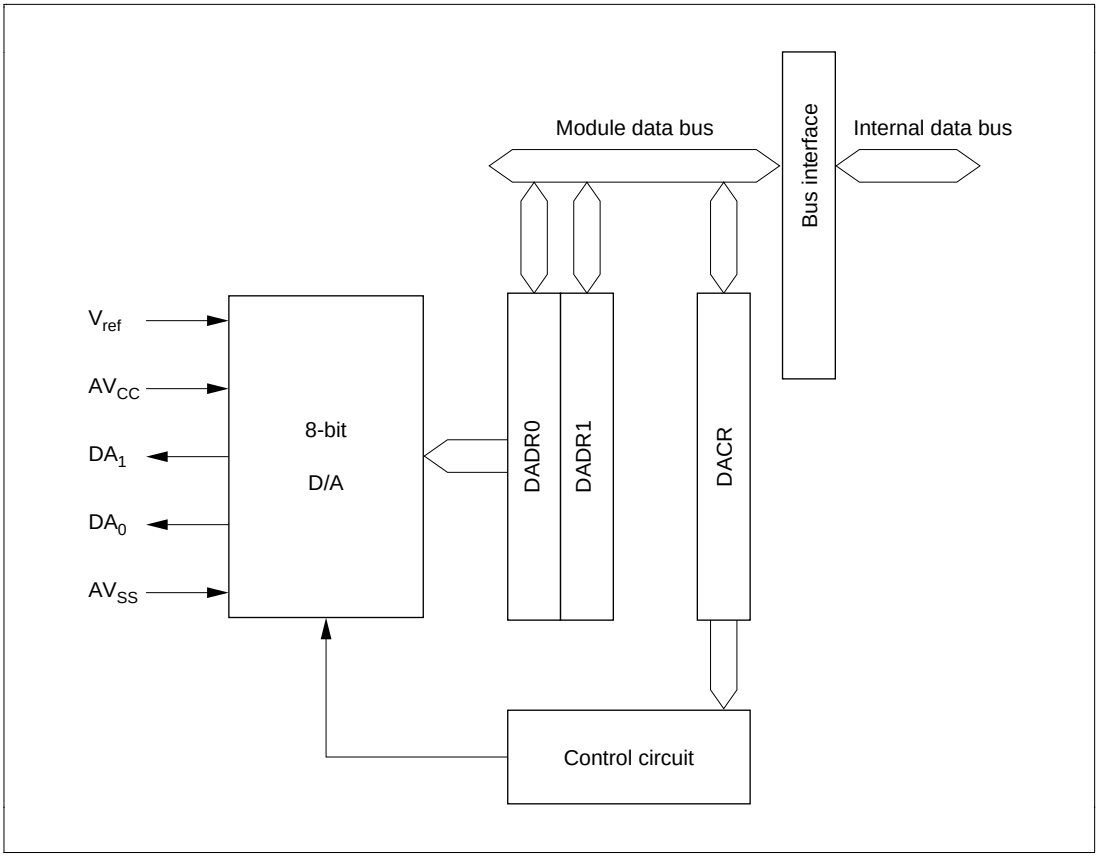



**Figure 17-1 Block Diagram of D/A Converter**

### 17.1.3 Pin Configuration

Table 17-1 summarizes the input and output pins of the D/A converter.

**Table 17-1 Pin Configuration**

| Pin Name | Symbol | I/O | Function |
|---|---|---|---|
| Analog power pin | $AV_{CC}$ | Input | Analog power source |
| Analog ground pin | $AV_{SS}$ | Input | Analog ground and reference voltage |
| Analog output pin 0 | $DA_0$ | Output | Channel 0 analog output |
| Analog output pin 1 | $DA_1$ | Output | Channel 1 analog output |
| Reference voltage pin | $V_{ref}$ | Input | Analog reference voltage |

### 17.1.4 Register Configuration

Table 17-2 summarizes the registers of the D/A converter.

**Table 17-2 D/A Converter Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| D/A data register 0 | DADR0 | R/W | H'00 | H'FFA4 |
| D/A data register 1 | DADR1 | R/W | H'00 | H'FFA5 |
| D/A control register | DACR | R/W | H'1F | H'FFA6 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Note:* Lower 16 bits of the address.

## 17.2 Register Descriptions

### 17.2.1 D/A Data Registers 0 and 1 (DADR0, DADR1)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DADR0 and DADR1 are 8-bit readable/writable registers that store data for conversion.

Whenever output is enabled, the values in DADR0 and DADR1 are converted and output from the analog output pins.

DADR0 and DADR1 are each initialized to H'00 by a reset and in hardware standby mode.

### 17.2.2 D/A Control Register (DACR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | DAOE1 | DAOE0 | DAE | — | — | — | — | — |
| Initial value: | | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | — | — | — | — | — |

DACR is an 8-bit readable/writable register that controls the operation of the D/A converter.

DACR is initialized to H'1F by a reset and in hardware standby mode.

**Bit 7—D/A Output Enable 1 (DAOE1):** Controls D/A conversion and analog output for channel 1.

**Bit 7**

| DAOE1 | Description | |
|---|---|---|
| 0 | Analog output $DA_1$ is disabled | (Initial value) |
| 1 | Channel 1 D/A conversion is enabled; analog output $DA_1$ is enabled | |

**Bit 6—D/A Output Enable 0 (DAOE0):** Controls D/A conversion and analog output for channel 0.

| Bit 6 | | |
|---|---|---|
| **DAOE0** | **Description** | |
| 0 | Analog output $DA_0$ is disabled | (Initial value) |
| 1 | Channel 0 D/A conversion is enabled; analog output $DA_0$ is enabled | |

**Bit 5—D/A Enable (DAE):** The DAOE0 and DAOE1 bits both control D/A conversion. When the DAE bit is cleared to 0, the channel 0 and 1 D/A conversions are controlled independently. When the DAE bit is set to 1, the channel 0 and 1 D/A conversions are controlled together.

Output of resultant conversions is always controlled independently by the DAOE0 and DAOE1 bits.

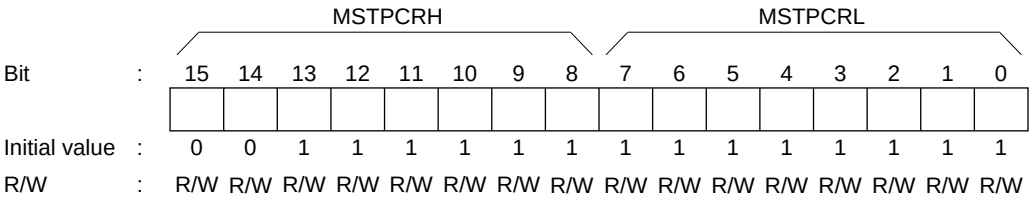
| Bit 7 | Bit 6 | Bit 5 | |
|---|---|---|---|
| **DAOE1** | **DAOE0** | **DAE** | **Description** |
| 0 | 0 | * | Channel 0 and 1 D/A conversions disabled |
| | 1 | 0 | Channel 0 D/A conversion enabled<br>Channel 1 D/A conversion disabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| 1 | 0 | 0 | Channel 0 D/A conversion disabled<br>Channel 1 D/A conversion enabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| | 1 | * | Channel 0 and 1 D/A conversions enabled |

*: Don't care

If the H8S/2655 Series enters software standby mode when D/A conversion is enabled, the D/A output is held and the analog power current is the same as during D/A conversion. When it is necessary to reduce the analog power current in software standby mode, clear both the DAOE0 and DAOE1 bits to 0 to disable D/A output.

**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

### 17.2.3 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP10 bit in MSTPCR is set to 1, D/A converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 21.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 10—Module Stop (MSTP10):** Specifies the D/A converter module stop mode.

**Bit 10**

| MSTP10 | Description | |
|---|---|---|
| 0 | D/A converter module stop mode cleared | |
| 1 | D/A converter module stop mode set | (Initial value) |

## 17.3 Operation

The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently.

D/A conversion is performed continuously while enabled by DACR. If either DADR0 or DADR1 is written to, the new data is immediately converted. The conversion result is output by setting the corresponding DAOE0 or DAOE1 bit to 1.

The operation example described in this section concerns D/A conversion on channel 0. Figure 17-2 shows the timing of this operation.

[1] Write the conversion data to DADR0.

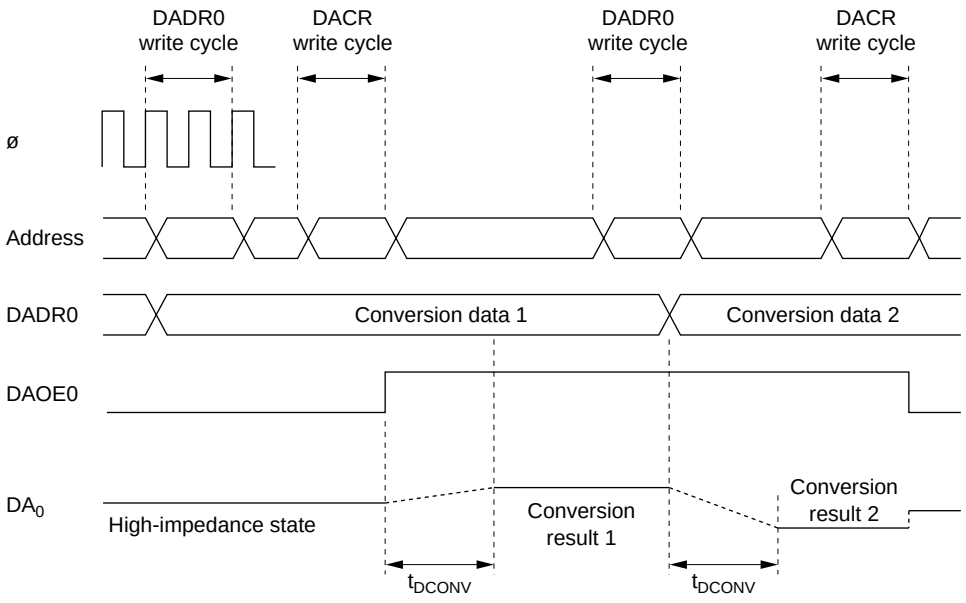
[2] Set the DAOE0 bit in DACR to 1. D/A conversion is started and the $DA_0$ pin becomes an output pin. The conversion result is output after the conversion time has elapsed. The output value is expressed by the following formula:

$$\frac{\text{DADR contents}}{256} \times V_{ref}$$

The conversion results are output continuously until DADR0 is written to again or the DAOE0 bit is cleared to 0.

[3] If DADR0 is written to again, the new data is immediately converted. The new conversion result is output after the conversion time has elapsed.

[4] If the DAOE0 bit is cleared to 0, the $DA_0$ pin becomes an input pin.




**Legend**

$t_{DCONV}$: D/A conversion time

**Figure 17-2 Example of D/A Converter Operation**

## 18.1 Overview

The H8S/2655 Series has 4 kbytes of on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data. This makes it possible to perform fast word data transfer.

The on-chip RAM can be enabled or disabled by means of the RAM enable bit (RAME) in the system control register (SYSCR).

### 18.1.1 Block Diagram

Figure 18-1 shows a block diagram of the on-chip RAM.

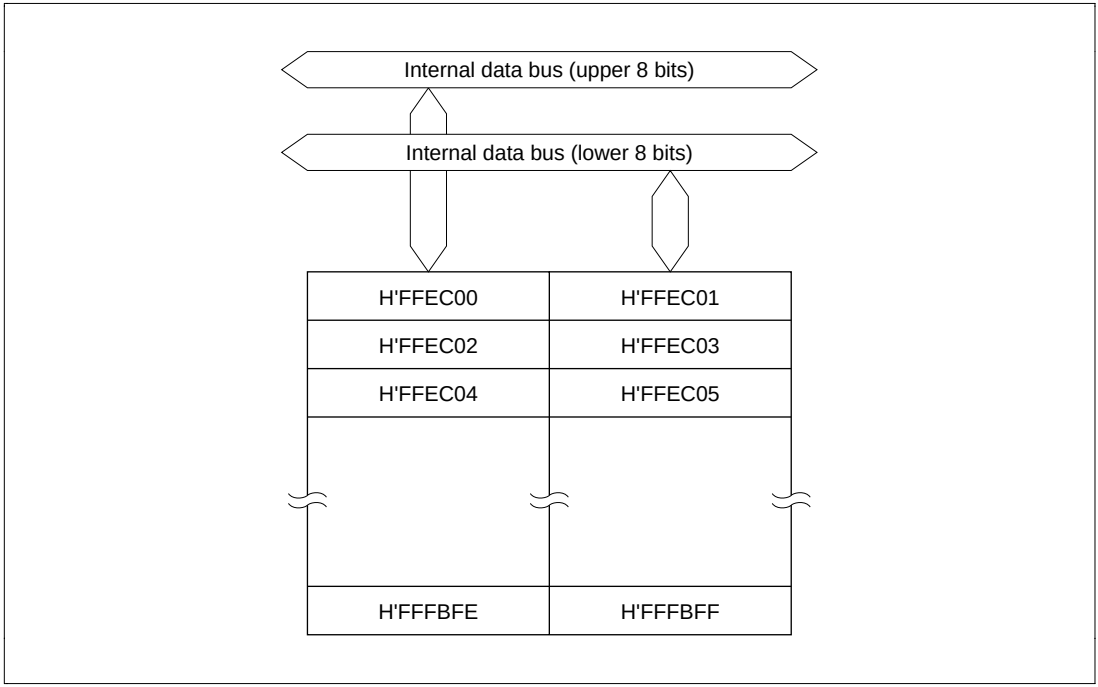



**Figure 18-1 Block Diagram of RAM**

### 18.1.2 Register Configuration

The on-chip RAM is controlled by SYSCR. Table 18-1 shows the address and initial value of SYSCR.

**Table 18-1 RAM Register**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| System control register | SYSCR | R/W | H'01 | H'FF39 |

Note: * Lower 16 bits of the address.

## 18.2 Register Descriptions

### 18.2.1 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | MACS | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | — | R/W | R/W | R/W | — | — | R/W |

The on-chip RAM is enabled or disabled by the RAME bit in SYSCR. For details of other bits in SYSCR, see section 3.2.2, System Control Register (SYSCR).

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized when the reset state is released. It is not initialized in software standby mode.

**Bit 0**

| RAME | Description | |
|---|---|---|
| 0 | On-chip RAM is disabled | |
| 1 | On-chip RAM is enabled | (Initial value) |

## 18.3 Operation

When the RAME bit is set to 1, accesses to addresses H'FFEC00 to H'FFFBFF are directed to the on-chip RAM. When the RAME bit is cleared to 0, the off-chip address space is accessed.

Since the on-chip RAM is connected to the CPU by an internal 16-bit data bus, it can be written to and read in byte or word units. Each type of access can be performed in one state.

Even addresses use the upper 8 bits, and odd addresses use the lower 8 bits. Word data must start at an even address.

## 18.4 Usage Notes

DTC register information can be located in addresses H'FFF800 to H'FFFBFF. When the DTC is used, the RAME bit must not be cleared to 0.

# Section 19 ROM

## 19.1 Overview

The H8S/2655 has 128 kbytes of on-chip ROM (PROM or mask ROM), and the H8S/2653 has 64 kbytes. The ROM is connected to the H8S/2600 CPU by a 16-bit data bus. The CPU accesses both byte data and word data in one state, making possible rapid instruction fetches and high-speed processing.

The on-chip ROM is enabled or disabled by setting the mode pins ($MD_2$, $MD_1$, and $MD_0$) and bit EAE in BCRL.

The PROM version of the H8S/2655 Series can be programmed with a general-purpose PROM programmer, by setting PROM mode.

### 19.1.1 Block Diagram

Figure 19-1 shows a block diagram of the on-chip ROM.

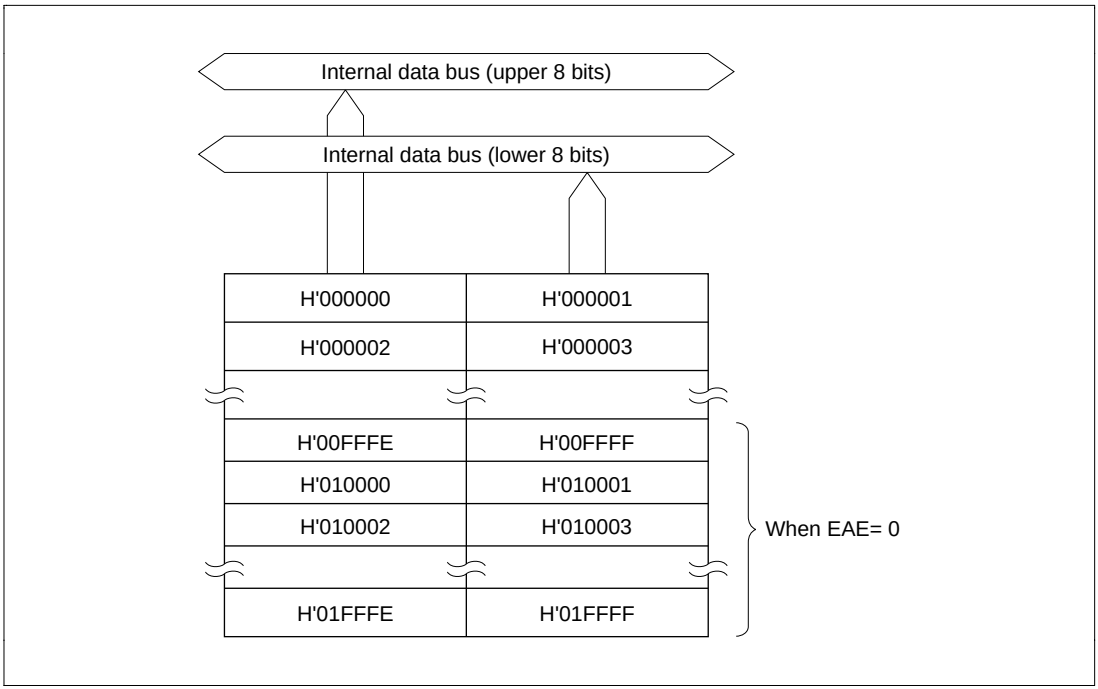



**Figure 19-1 Block Diagram of ROM (H8S/2655)**

### 19.1.2 Register Configuration

The H8S/2655's on-chip ROM is controlled by BCRL. The register configuration is shown in table 19-1.

**Table 19-1 ROM Register**

| | | | Initial Value | | |
|---|---|---|---|---|---|
| **Name** | **Abbreviation** | **R/W** | **Power-On Reset** | **Manual Reset** | **Address*** |
| Bus control register L | BCRL | R/W | H'3C | Retained | H'FED5 |

Note: * Lower 16 bits of the address.

## 19.2 Register Descriptions

### 19.2.1 Bus Control Register L (BCRL)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | BRLE | BREQOE | EAE | LCASS | DDS | ASS | WDBE | WAITE |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Enabling or disabling of part of the H8S/2655's on-chip ROM area can be selected by means of the EAE bit in BCRL. For details of the other bits in BCRL, see 6.2.5, Bus Control Register L.

**Bit 5—External Address Enable (EAE):** Selects whether addresses H'010000 to H'01FFFF are to be internal addresses or external addresses.

This setting is invalid in normal mode.

**Bit 5**

| EAE | Description |
|---|---|
| 0 | Addresses H'010000 to H'01FFFF are in on-chip ROM (in the H8S/2655) or a reserved area* (in the H8S/2653). |
| 1 | Addresses H'010000 to H'01FFFF are external addresses (external expansion mode) or a reserved area* (single-chip mode). (Initial value) |

Note: * Reserved areas should not be accessed.

## 19.3 Operation

The on-chip ROM is connected to the CPU by a 16-bit data bus, and both byte and word data can be accessed in one state. Even addresses are connected to the upper 8 bits, and odd addresses to the lower 8 bits. Word data must start at an even address.

The on-chip ROM is enabled and disabled by setting the mode pins ($MD_2$, $MD_1$, and $MD_0$) and bit EAE in BCRL. These settings are shown in table 19-2.

In normal mode, a maximum of 56 kbytes of ROM can be used.

**Table 19-2 Operating Modes and ROM Area**

| | | Mode Pin | | | BCRL | |
|---|---|---|---|---|---|---|
| **Operating Mode** | | **MD2** | **MD1** | **MD0** | **EAE** | **On-Chip ROM** |
| Mode 1 | Normal expanded mode with on-chip ROM disabled | 0 | 0 | 1 | — | Disabled |
| Mode 2 | Normal expanded mode with on-chip ROM enabled | | 1 | 0 | — | Enabled (56 kbytes) |
| Mode 3 | Normal single-chip mode | | | 1 | | |
| Mode 4 | Advanced expanded mode with on-chip ROM disabled | 1 | 0 | 0 | — | Disabled |
| Mode 5 | Advanced expanded mode with on-chip ROM disabled | | | 1 | | |
| Mode 6 | Advanced expanded mode with on-chip ROM enabled | | 1 | 0 | 0 | Enabled* |
| | | | | | 1 | Enabled (64 kbytes) |
| Mode 7 | Advanced single-chip mode | | | 1 | 0 | Enabled* |
| | | | | | 1 | Enabled (64 kbytes) |

Note: * 128 kbytes in the H8S/2655, 64 kbytes in the H8S/2653
In H8/2655 modes 6 and 7, the on-chip ROM available after a power-on reset is the 64-kbyte area comprising addresses H'000000 to H'00FFFF.

## 19.4 PROM Mode

### 19.4.1 PROM Mode Setting

The PROM version of the H8S/2655 Series suspends its microcontroller functions when placed in PROM mode, enabling the on-chip PROM to be programmed. This programming can be done with a PROM programmer set up in the same way as for the HN27C101 EPROM ($V_{PP}$ = 12.5 V). Use of a 120-pin/32-pin socket adapter enables programming with a commercial PROM programmer.

Note that the PROM programmer should not be set to page mode as the H8S/2655 Series does not support page programming.

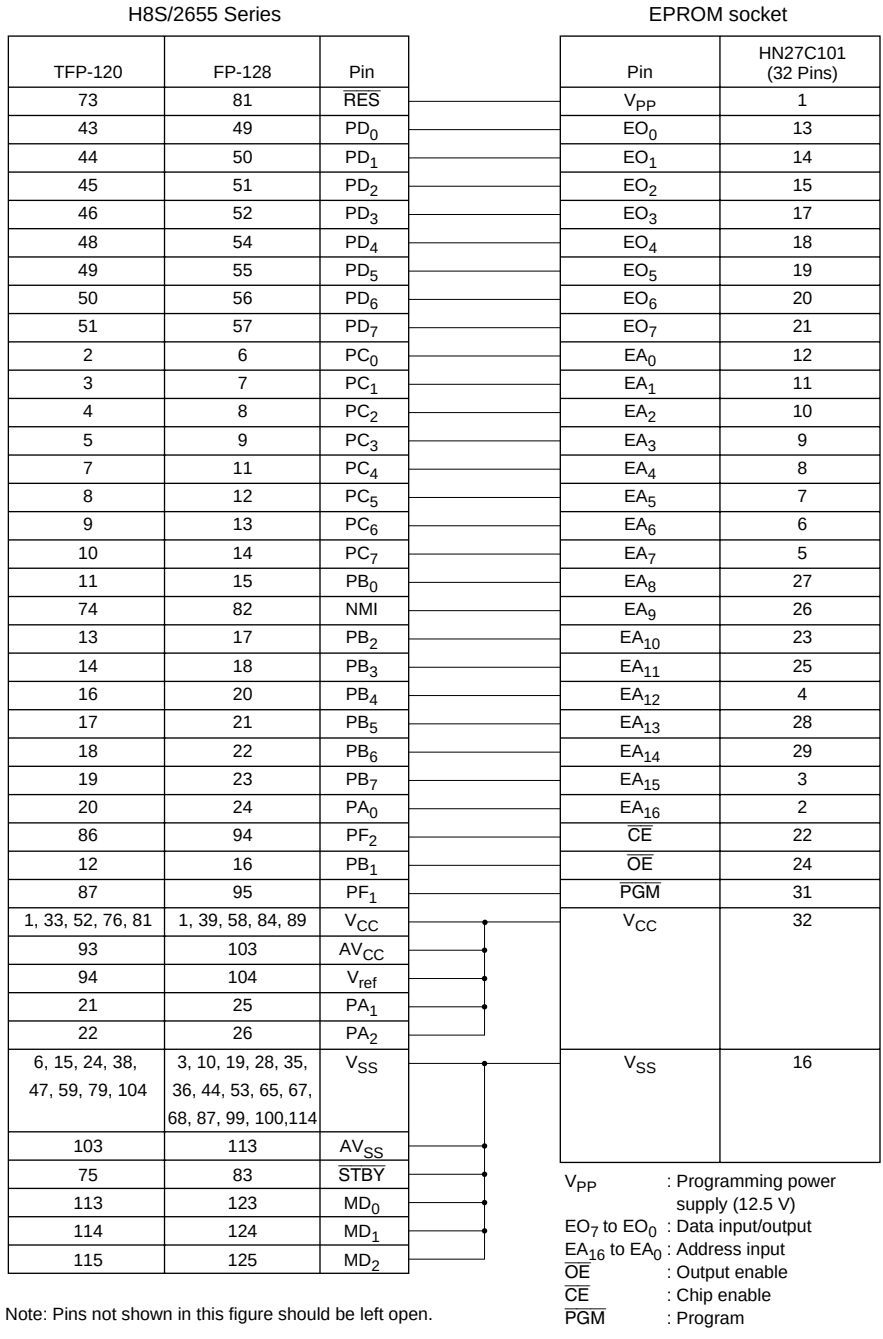
Table 19-3 shows how PROM mode is selected.

**Table 19-3 Selecting PROM Mode**

| Pin Names | Setting |
|---|---|
| $MD_2$, $MD_1$, $MD_0$ | Low |
| $\overline{STBY}$ | |
| $PA_2$, $PA_1$ | High |

### 19.4.2 Socket Adapter and Memory Map

Programs can be written and verified by attaching a 120-pin/32-pin socket adapter to the PROM programmer. Table 19-4 gives ordering information for the socket adapter, and figure 19-2 shows the wiring of the socket adapter. Figure 19-3 shows the memory map in PROM mode.




Note: Pins not shown in this figure should be left open.

**Figure 19-2 Wiring of 120-Pin/32-Pin Socket Adapter**

**Table 19-4 Socket Adapter**

| Microcontroller | Package | Socket Adapter |
|---|---|---|
| H8S/2655 | 120 pin TQFP (TFP-120) | HS2655ESNS1H |
| | 128 pin QFP (FP-128) | HS2655ESHS1H |

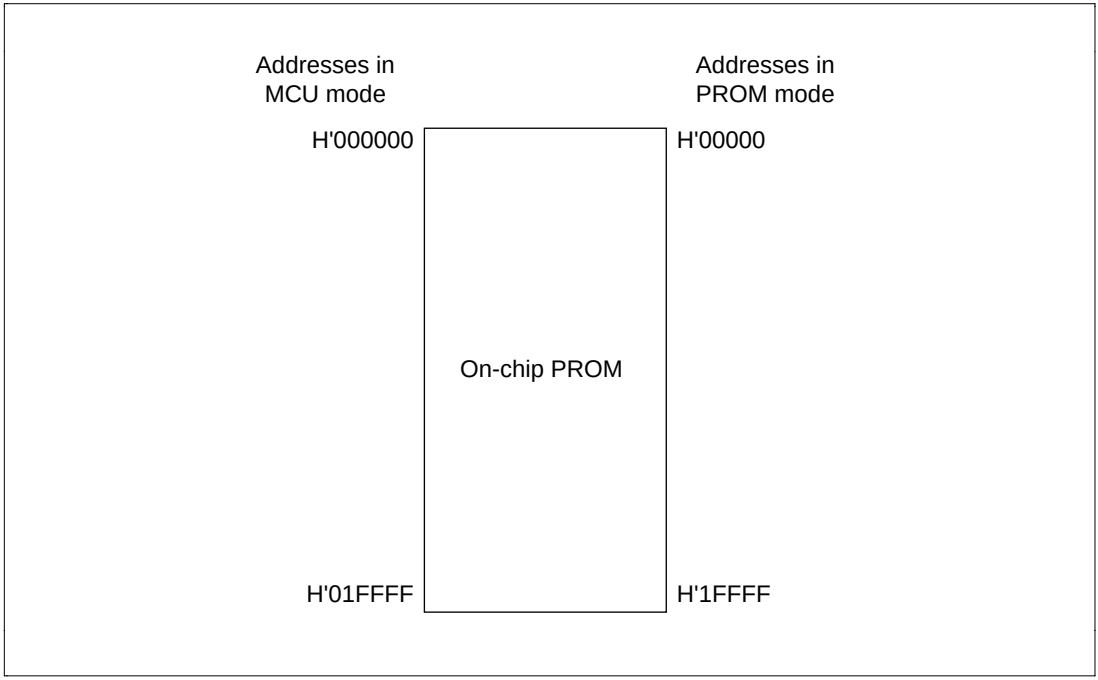



**Figure 19-3 Memory Map in PROM Mode**

## 19.5 Programming

### 19.5.1 Overview

Table 19-5 shows how to select the program, verify, and program-inhibit modes in PROM mode.

**Table 19-5 Mode Selection in PROM Mode**

| | Pins | | | | | | |
|---|---|---|---|---|---|---|---|
| **Mode** | **$\overline{CE}$** | **$\overline{OE}$** | **$\overline{PGM}$** | **$V_{PP}$** | **$V_{CC}$** | **$EO_7$ to $EO_0$** | **$EA_{16}$ to $EA_0$** |
| Program | L | H | L | $V_{PP}$ | $V_{CC}$ | Data input | Address input |
| Verify | L | L | H | $V_{PP}$ | $V_{CC}$ | Data output | Address input |
| Program-inhibit | L | L | L | $V_{PP}$ | $V_{CC}$ | High impedance | Address input |
| | L | H | H | | | | |
| | H | L | L | | | | |
| | H | H | H | | | | |

Legend

L : Low voltage level
H : High voltage level
$V_{PP}$ : $V_{PP}$ voltage level
$V_{CC}$ : $V_{CC}$ voltage level

Programming and verification should be carried out using the same specifications as for the standard HN27C101 EPROM.

However, do not set the PROM programmer to page mode does not support page programming. A PROM programmer that only supports page programming cannot be used. When choosing a PROM programmer, check that it supports high-speed programming in byte units. Always set addresses within the range H'00000 to H'1FFFF.

### 19.5.2 Programming and Verification

An efficient, high-speed programming procedure can be used to program and verify PROM data. This procedure writes data quickly without subjecting the chip to voltage stress or sacrificing data reliability. It leaves the data H'FF in unused addresses. Figure 19-4 shows the basic high-speed programming flowchart. Tables 19-6 and 19-7 list the electrical characteristics of the chip during programming. Figure 19-5 shows a timing chart.

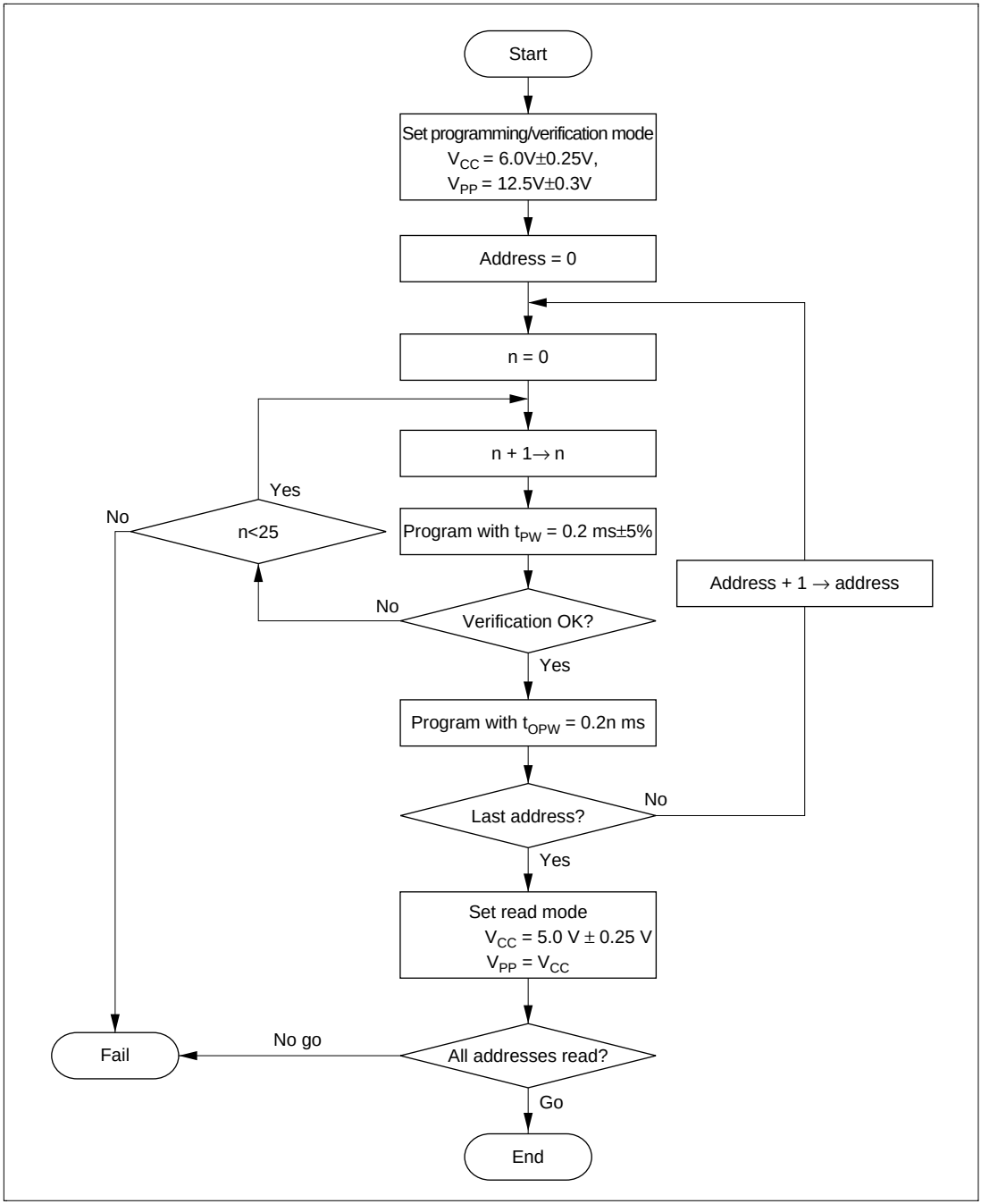



**Figure 19-4 High-Speed Programming Flowchart**

**Table 19-6 DC Characteristics in PROM Mode**

(When $V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 V ± 0.3 V, $V_{SS}$ = 0 V, $T_a$ = 25°C ± 5°C)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Input high voltage | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $V_{IH}$ | 2.4 | — | $V_{CC}$ + 0.3 | V | |
| Input low voltage | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $V_{IL}$ | –0.3 | — | 0.8 | V | |
| Output high voltage | $EO_7$ to $EO_0$ | $V_{OH}$ | 2.4 | — | — | V | $I_{OH}$ = –200 µA |
| Output low voltage | $EO_7$ to $EO_0$ | $V_{OL}$ | — | — | 0.45 | V | $I_{OL}$ = 1.6 mA |
| Input leakage current | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $\lvert I_{LI} \rvert$ | — | — | 2 | µA | $V_{in}$ = 5.25 V/0.5 V |
| $V_{CC}$ current | | $I_{CC}$ | — | — | 40 | mA | |
| $V_{PP}$ current | | $I_{PP}$ | — | — | 40 | mA | |

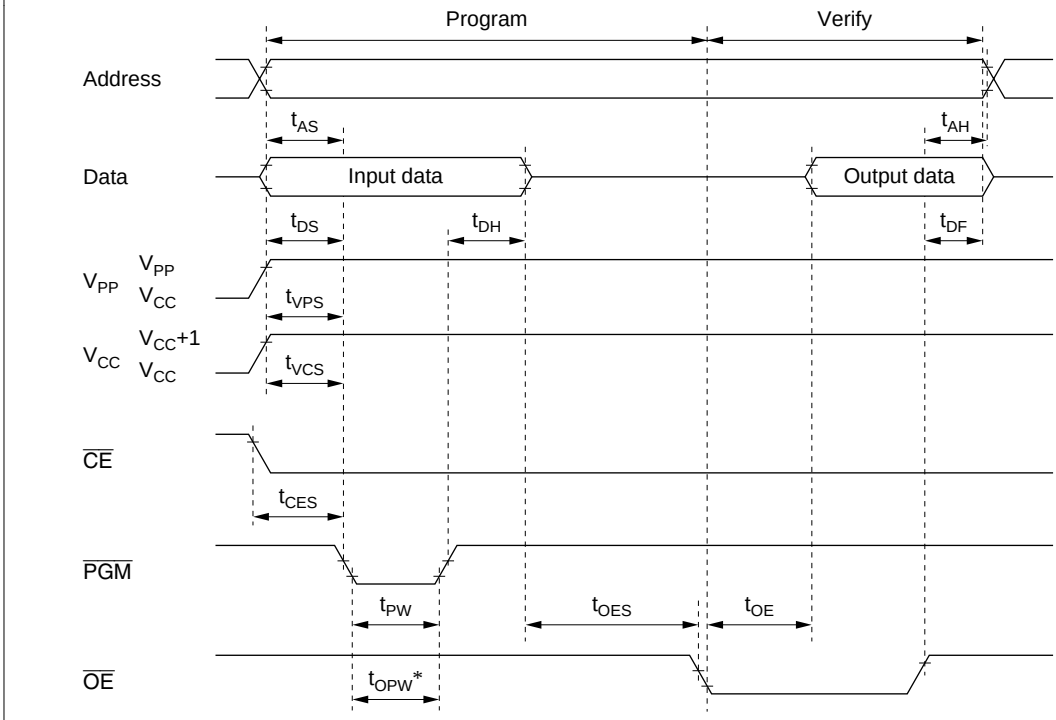
**Table 19-7 AC Characteristics in PROM Mode**

(When $V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 V ± 0.3 V, $T_a$ = 25°C ± 5°C)

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| Address setup time | $t_{AS}$ | 2 | — | — | µs | Figure 19-5*[1] |
| $\overline{OE}$ setup time | $t_{OES}$ | 2 | — | — | µs | |
| Data setup time | $t_{DS}$ | 2 | — | — | µs | |
| Address hold time | $t_{AH}$ | 0 | — | — | µs | |
| Data hold time | $t_{DH}$ | 2 | — | — | µs | |
| Data output disable time | $t_{DF}$*[2] | — | — | 130 | ns | |
| $V_{PP}$ setup time | $t_{VPS}$ | 2 | — | — | µs | |
| Programming pulse width | $t_{PW}$ | 0.19 | 0.20 | 0.21 | ms | |
| $\overline{PGM}$ pulse width for overwrite programming | $t_{OPW}$*[3] | 0.19 | — | 5.25 | ms | |
| $V_{CC}$ setup time | $t_{VCS}$ | 2 | — | — | µs | |
| $\overline{CE}$ setup time | $t_{CES}$ | 2 | — | — | µs | |
| Data output delay time | $t_{OE}$ | 0 | — | 150 | ns | |

Notes: 1. Input pulse level: 0.8 V to 2.2 V
Input rise time and fall time ≤ 20 ns
Timing reference levels: Input: 1.0 V, 2.0 V
Output: 0.8 V, 2.0 V

2. $t_{DF}$ is defined to be when output has reached the open state, and the output level can no longer be referenced.
3. $t_{OPW}$ is defined by the value shown in the flowchart.




Note: * $t_{OPW}$ is defined by the value shown in the flowchart.

**Figure 19-5 PROM Programming/Verification Timing**

### 19.5.3 Programming Precautions

- Program using the specified voltages and timing.

  The programming voltage ($V_{PP}$) in PROM mode is 12.5 V.

  If the PROM programmer is set to Hitachi HN27C101 specifications, $V_{PP}$ will be 12.5 V. Applied voltages in excess of the specified values can permanently destroy the MCU. Be particularly careful about the PROM programmer's overshoot characteristics.

- Before programming, check that the MCU is correctly mounted in the PROM programmer. Overcurrent damage to the MCU can result if the index marks on the PROM programmer, socket adapter, and MCU are not correctly aligned.

- Do not touch the socket adapter or MCU while programming. Touching either of these can cause contact faults and programming errors.

- The MCU cannot be programmed in page programming mode. Select the programming mode carefully.

- The size of the H8S/2655 PROM is 128 kbytes. Always set addresses within the range H'00000 to H'1FFFF. During programming, write H'FF to unused addresses to avoid verification errors.

### 19.5.4 Reliability of Programmed Data

An effective way to assure the data retention characteristics of the programmed chips is to bake them at 150°C, then screen them for data errors. This procedure quickly eliminates chips with PROM memory cells prone to early failure.

Figure 19-6 shows the recommended screening procedure.

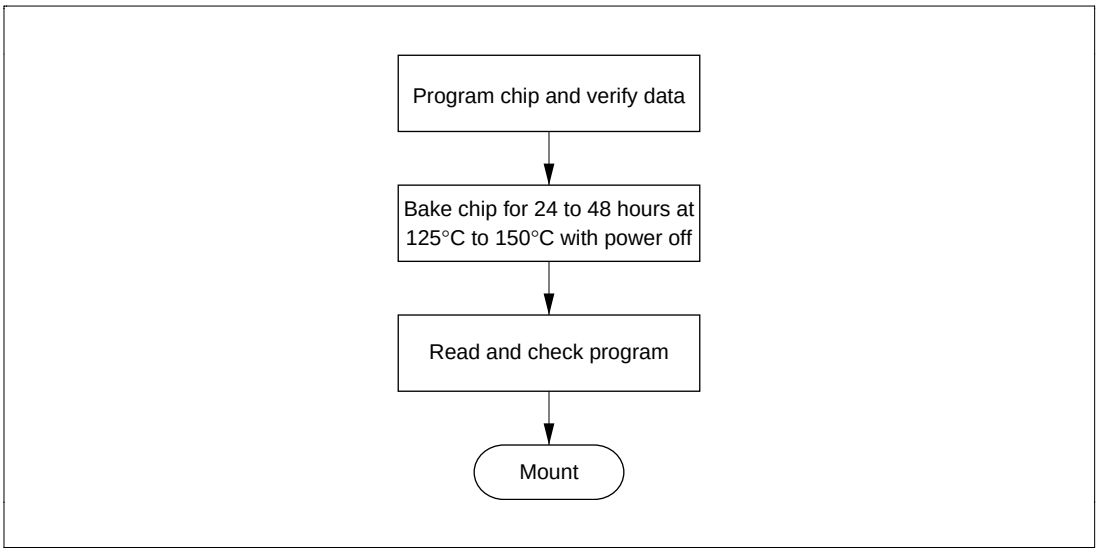



**Figure 19-6 Recommended Screening Procedure**

If a series of programming errors occurs while the same PROM programmer is being used, stop programming and check the PROM programmer and socket adapter for defects.

Please inform Hitachi of any abnormal conditions noted during or after programming or in screening of program data after high-temperature baking.

## 20.1 Overview

The H8S/2655 Series has a built-in clock pulse generator (CPG) that generates the system clock (ø), the bus master clock, and internal clocks.

The clock pulse generator consists of an oscillator circuit, a duty adjustment circuit, a medium-speed clock divider, and a bus master clock selection circuit.

### 20.1.1 Block Diagram

Figure 20-1 shows a block diagram of the clock pulse generator.

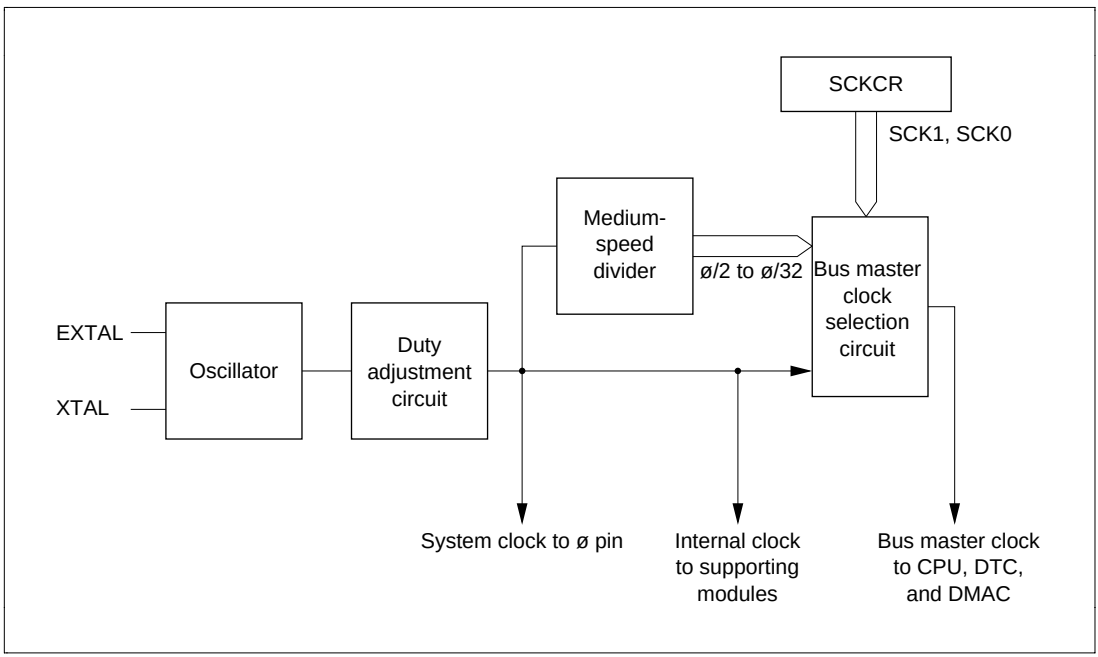



**Figure 20-1 Block Diagram of Clock Pulse Generator**

### 20.1.2 Register Configuration

The clock pulse generator is controlled by SCKCR. Table 20-1 shows the register configuration.

**Table 20-1 Clock Pulse Generator Register**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| System clock control register | SCKCR | R/W | H'00 | H'FF3A |

Note:* Lower 16 bits of the address.

## 20.2 Register Descriptions

### 20.2.1 System Clock Control Register (SCKCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | — | — | — | R/W | R/W | R/W |

SCKCR is an 8-bit readable/writable register that performs ø clock output control and medium-speed mode control.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—ø Clock Output Disable (PSTOP):** Controls ø output.

| | Description | | | |
|---|---|---|---|---|
| **Bit 7 PSTOP** | **Normal Operation** | **Sleep Mode** | **Software Standby Mode** | **Hardware Standby Mode** |
| 0 | ø output (initial value) | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

**Bit 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bits 5 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select 2 to 0 (SCK2 to SCK0):** These bits select the clock for the bus master.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| **SCK2** | **SCK1** | **SCK0** | **Description** | |
| 0 | 0 | 0 | Bus master is in high-speed mode | (Initial value) |
| | | 1 | Medium-speed clock is ø/2 | |
| | 1 | 0 | Medium-speed clock is ø/4 | |
| | | 1 | Medium-speed clock is ø/8 | |
| 1 | 0 | 0 | Medium-speed clock is ø/16 | |
| | | 1 | Medium-speed clock is ø/32 | |
| | 1 | — | — | |

## 20.3 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

### 20.3.1 Connecting a Crystal Resonator

**Circuit Configuration:** A crystal resonator can be connected as shown in the example in figure 20-2. Select the damping resistance $R_d$ according to table 20-2. An AT-cut parallel-resonance crystal should be used.

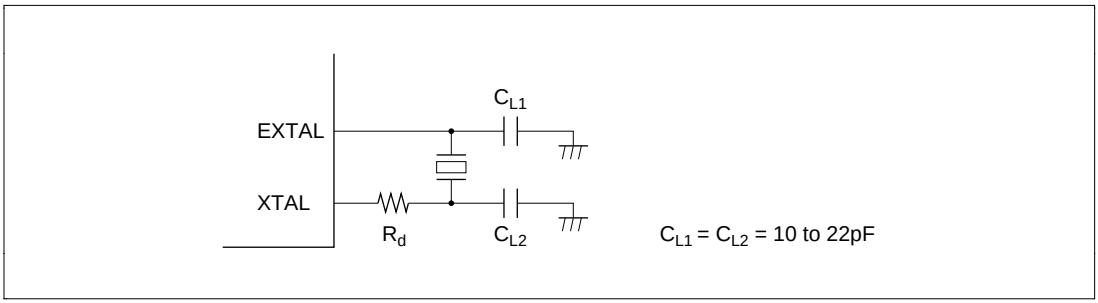



**Figure 20-2 Connection of Crystal Resonator (Example)**

**Table 20-2 Damping Resistance Value**

| Frequency (MHz) | 2 | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| $R_d$ (Ω) | 1k | 500 | 200 | 0 | 0 | 0 |

**Crystal Resonator:** Figure 20-3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 20-3 and the same resonance frequency as the system clock (ø).

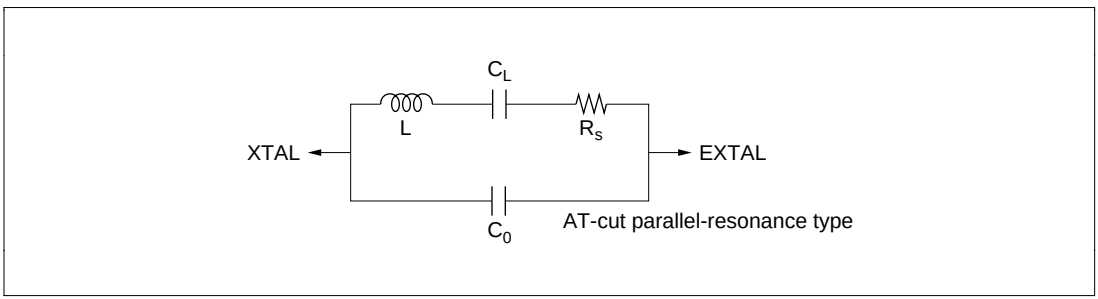



**Figure 20-3 Crystal Resonator Equivalent Circuit**

**Table 20-3 Crystal Resonator Parameters**

| Frequency (MHz) | 2 | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| $R_S$ max (Ω) | 500 | 120 | 80 | 60 | 50 | 40 |
| $C_0$ max (pF) | 7 | 7 | 7 | 7 | 7 | 7 |

**Note on Board Design:** When a crystal resonator is connected, the following points should be noted:

Other signal lines should be routed away from the oscillator circuit to prevent induction from interfering with correct oscillation. See figure 20-4.

When designing the board, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins.

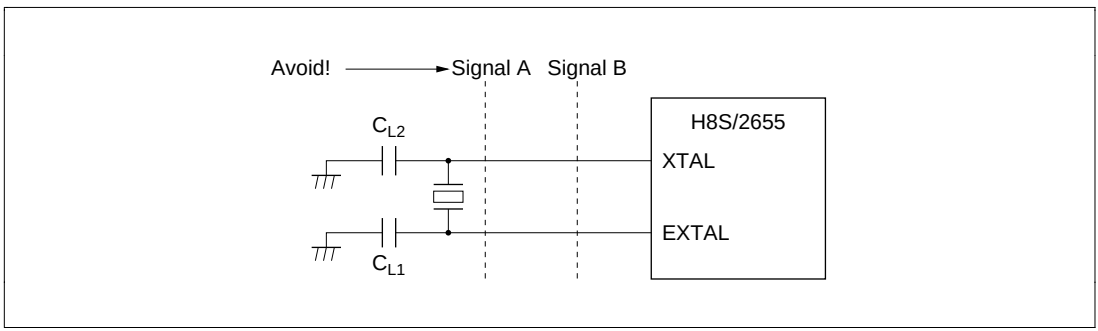



**Figure 20-4 Example of Incorrect Board Design**

### 20.3.2 External Clock Input

**Circuit Configuration:** An external clock signal can be input as shown in the examples in figure 20-5. If the XTAL pin is left open, make sure that stray capacitance is no more than 10 pF.

In example (b), make sure that the external clock is held high in standby mode.

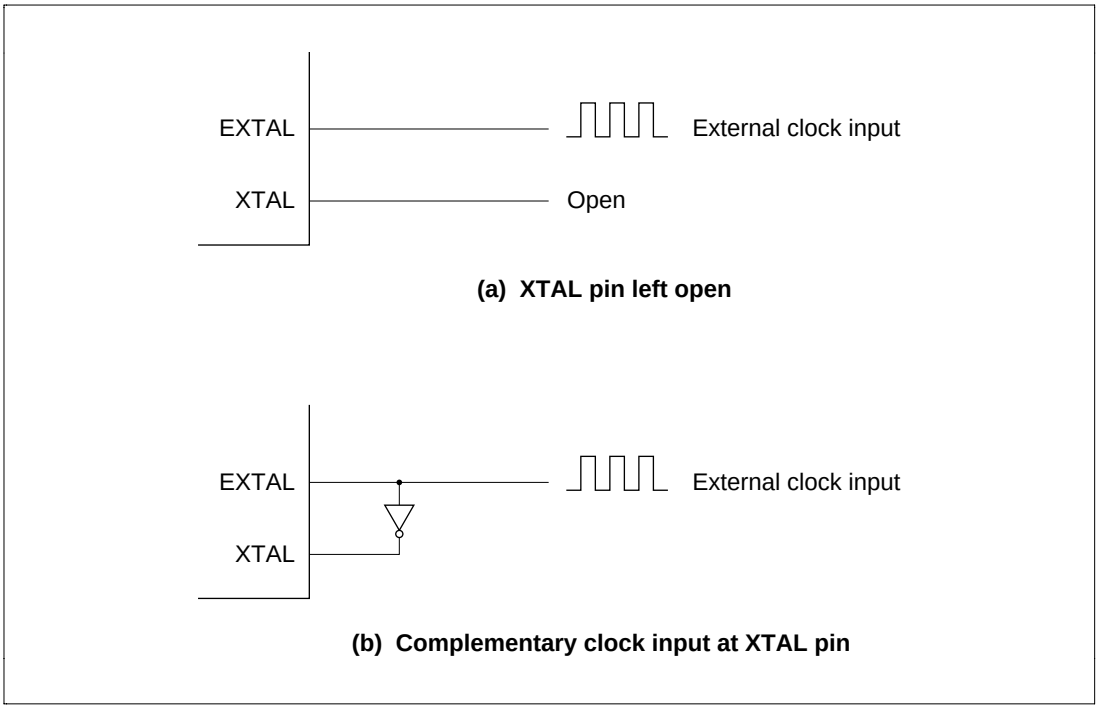



**Figure 20-5 External Clock Input (Examples)**

**External Clock:** The external clock signal should have the same frequency as the system clock (ø).

Table 20-4 and figure 20-6 show the input conditions for the external clock.

**Table 20-4 External Clock Input Conditions**

| Item | Symbol | $V_{cc}$ = 2.7 V to 5.5 V Min | Max | $V_{cc}$ = 5.0 V ± 10% Min | Max | Unit | Test Conditions | |
|---|---|---|---|---|---|---|---|---|
| External clock input low pulse width | $t_{EXL}$ | 40 | — | 20 | — | ns | Figure 20-6 | |
| External clock input high pulse width | $t_{EXH}$ | 40 | — | 20 | — | ns | | |
| External clock rise time | $t_{EXr}$ | — | 10 | — | 5 | ns | | |
| External clock fall time | $t_{EXf}$ | — | 10 | — | 5 | ns | | |
| Clock low pulse width level | $t_{CL}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{cyc}$ | ø ≥ 5 MHz | Figure 22-4 |
| | | 80 | — | 80 | — | ns | ø < 5 MHz | |
| Clock high pulse width level | $t_{CH}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{cyc}$ | ø ≥ 5 MHz | |
| | | 80 | — | 80 | — | ns | ø < 5 MHz | |

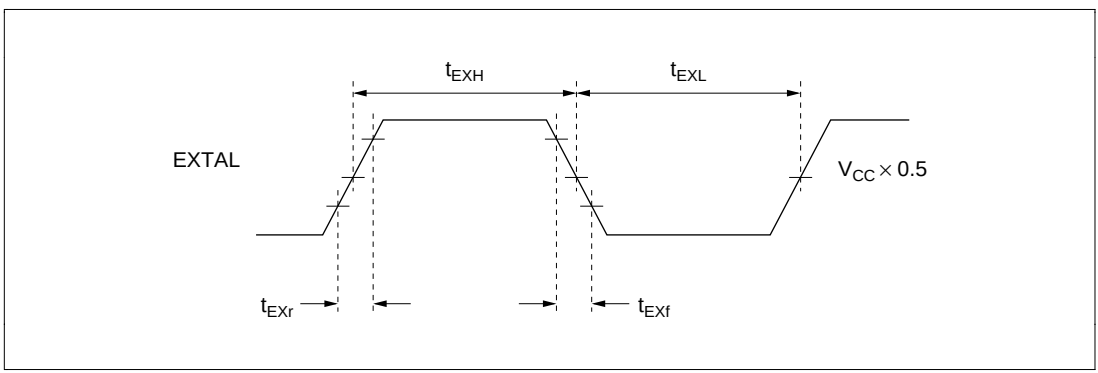



**Figure 20-6 External Clock Input Timing**

## 20.4 Duty Adjustment Circuit

When the oscillator frequency is 5 MHz or higher, the duty adjustment circuit adjusts the duty cycle of the clock signal from the oscillator to generate the system clock (ø).

## 20.5 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock to generate ø/2, ø/4, ø/8, ø/16, and ø/32.

## 20.6 Bus Master Clock Selection Circuit

The bus master clock selection circuit selects the system clock (ø) or one of the medium-speed clocks (ø/2, ø/4, or ø/8, ø/16, and ø/32) to be supplied to the bus master, according to the settings of the SCK2 to SCK0 bits in SCKCR.

# Section 21 Power-Down Modes

## 21.1 Overview

In addition to the normal program execution state, the H8S/2655 Series has five power-down modes in which operation of the CPU and oscillator is halted and power dissipation is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

The H8S/2655 Series operating modes are as follows:

(1) High-speed mode
(2) Medium-speed mode
(3) Sleep mode
(4) Module stop mode
(5) Software standby mode
(6) Hardware standby mode

Of these, (2) to (6) are power-down modes. Sleep mode is a CPU mode, medium-speed mode is a CPU and bus master mode, and module stop mode is an on-chip supporting module mode (including bus masters other than the CPU). A combination of these modes can be set.

After a reset, the H8S/2655 Series is in high-speed mode.

Table 21-1 shows the conditions for transition to the various modes, the status of the CPU, on-chip supporting modules, etc., and the method of clearing each mode.

**Table 21-1 Operating Modes**

| Operating Mode | Transition Condition | Clearing Condition | Oscillator | CPU | | Modules | | I/O Ports |
|---|---|---|---|---|---|---|---|---|
| | | | | | Registers | | Registers | |
| High speed mode | Control register | | Functions | High speed | Functions | High speed | Functions | High speed |
| Medium-speed mode | Control register | | Functions | Medium speed | Functions | High/ medium speed *1 | Functions | High speed |
| Sleep mode | Instruction | Interrupt | Functions | Halted | Retained | High speed | Functions | High speed |
| Module stop mode | Control register | | Functions | High/ medium speed | Functions | Halted | Retained/ reset *2 | Retained |
| Software standby mode | Instruction | External interrupt | Halted | Halted | Retained | Halted | Retained/ reset *2 | Retained |
| Hardware standby mode | Pin | | Halted | Halted | Undefined | Halted | Reset | High impedance |

Notes: 1. The bus master operates on the medium-speed clock, and other on-chip supporting modules on the high-speed clock.
2. The SCI is reset, and other on-chip supporting modules retain their state.

### 21.1.1 Register Configuration

Power-down modes are controlled by the SBYCR, SCKCR, and MSTPCR registers. Table 21-2 summarizes these registers.

**Table 21-2 Power-Down Mode Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Standby control register | SBYCR | R/W | H'08 | H'FF38 |
| System clock control register | SCKCR | R/W | H'00 | H'FF3A |
| Module stop control register H | MSTPCRH | R/W | H'3F | H'FF3C |
| Module stop control register L | MSTPCRL | R/W | H'FF | H'FF3D |

Note: * Lower 16 bits of the address.

## 21.2 Register Descriptions

### 21.2.1 Standby Control Register (SBYCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | — | — | — |

SBYCR is an 8-bit readable/writable register that performs software standby mode control.

SBYCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Software Standby (SSBY):** Specifies a transition to software standby mode. Remains set to 1 when software standby mode is released by an external interrupt, and a transition is made to normal operation. The SSBY bit should be cleared by writing 0 to it.

| Bit 7 | | |
|---|---|---|
| **SSBY** | **Description** | |
| 0 | Transition to sleep mode after execution of SLEEP instruction | (Initial value) |
| 1 | Transition to software standby mode after execution of SLEEP instruction | |

**Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0):** These bits select the time the MCU waits for the clock to stabilize when software standby mode is cleared by an external interrupt. With crystal oscillation, refer to table 21-4 and make a selection according to the operating frequency so that the standby time is at least 8 ms (the oscillation stabilization time). With an external clock, any selection can be made.

| Bit 6 | Bit 5 | Bit 4 | | |
|---|---|---|---|---|
| **STS2** | **STS1** | **STS0** | **Description** | |
| 0 | 0 | 0 | Standby time = 8192 states | (Initial value) |
| | | 1 | Standby time = 16384 states | |
| | 1 | 0 | Standby time = 32768 states | |
| | | 1 | Standby time = 65536 states | |
| 1 | 0 | 0 | Standby time = 131072 states | |
| | | 1 | Standby time = 262144 states | |
| | 1 | 0 | Reserved | |
| | | 1 | Standby time = 16 states | |

**Bit 3—Output Port Enable (OPE):** Specifies whether the output of the address bus and bus control signals ($\overline{CS_0}$ to $\overline{CS_7}$, $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, $\overline{CAS}$, $\overline{OE}$) is retained or set to the high-impedance state in software standby mode.

| Bit 3 OPE | Description |
|---|---|
| 0 | In software standby mode, address bus and bus control signals are high-impedance |
| 1 | In software standby mode, address bus and bus control signals retain output state (Initial value) |

**Bits 2 to 0—Reserved:** Read-only bits, always read as 0.

### 21.2.2 System Clock Control Register (SCKCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | — | — | — | R/W | R/W | R/W |

SCKCR is an 8-bit readable/writable register that performs ø clock output control and medium-speed mode control.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—ø Clock Output Disable (PSTOP):** Controls ø output.

| Bit 7 PSTOP | Description: Normal Operating Mode | Sleep Mode | Software Standby Mode | Hardware Standby Mode |
|---|---|---|---|---|
| 0 | ø output (initial value) | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

**Bits 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bits 5 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select (SCK2 to SCK0):** These bits select the clock for the bus master.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| **SCK2** | **SCK1** | **SCK0** | **Description** | |
| 0 | 0 | 0 | Bus master in high-speed mode | (Initial value) |
| | | 1 | Medium-speed clock is ø/2 | |
| | 1 | 0 | Medium-speed clock is ø/4 | |
| | | 1 | Medium-speed clock is ø/8 | |
| 1 | 0 | 0 | Medium-speed clock is ø/16 | |
| | | 1 | Medium-speed clock is ø/32 | |
| | 1 | — | — | |

### 21.2.3 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 15 to 0—Module Stop (MSTP 15 to MSTP 0):** These bits specify module stop mode. See table 21-3 for the method of selecting on-chip supporting modules.

| Bits 15 to 0 | |
|---|---|
| **MSTP15 to MSTP0** | **Description** |
| 0 | Module stop mode cleared |
| 1 | Module stop mode set |

## 21.3 Medium-Speed Mode

When the SCK2 to SCK0 bits in SCKCR are set to 1, the operating mode changes to medium-speed mode as soon as the current bus cycle ends. In medium-speed mode, the CPU operates on the operating clock (ø/2, ø/4, ø/8, ø/16, or ø/32) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (the DMAC and DTC) also operate in medium-speed mode. On-chip supporting modules other than the bus masters always operate on the high-speed clock (ø).

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if ø/4 is selected as the operating clock, on-chip memory is accessed in 4 states, and internal I/O registers in 8 states.

Medium-speed mode is cleared by clearing all of bits SCK2 to SCK0 to 0. A transition is made to high-speed mode and medium-speed mode is cleared at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, a transition is made to software standby mode. When software standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the $\overline{\text{RES}}$ pin is driven low, a transition is made to the reset state, and medium-speed mode is cleared. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Figure 21-1 shows the timing for transition to and clearance of medium-speed mode.

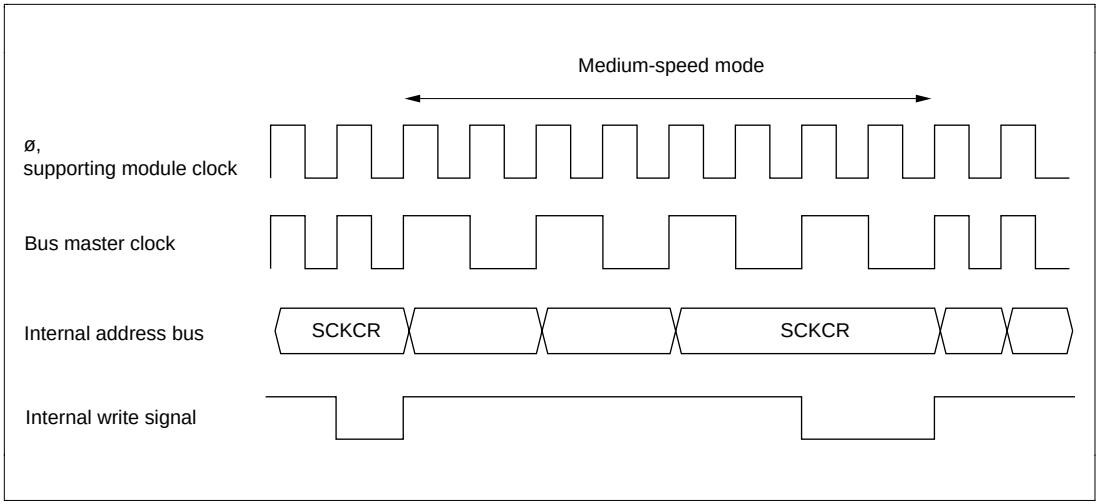



**Figure 21-1 Medium-Speed Mode Transition and Clearance Timing**

## 21.4 Sleep Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting modules do not stop.

Sleep mode is cleared by a reset or any interrupt, and the CPU returns to the normal program execution state via the exception handling state. Sleep mode is not cleared if interrupts are disabled, or if interrupts other than NMI are masked by the CPU.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

## 21.5 Module Stop Mode

### 21.5.1 Module Stop Mode

Module stop mode can be set for individual on-chip supporting modules.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

Table 21-3 shows MSTP bits and the corresponding on-chip supporting modules.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI are retained.

After reset clearance, all modules other than DMAC and DTC are in module stop mode.

When an on-chip supporting module is in module stop mode, read/write access to its registers is disabled.

**Table 21-3 MSTP Bits and Corresponding On-Chip Supporting Modules**

| Register | Bit | Module |
|---|---|---|
| MSTPCRH | MSTP15 | DMA controller |
| | MSTP14 | Data transfer controller (DTC) |
| | MSTP13 | 16-bit timer pulse unit (TPU) |
| | MSTP12 | 8-bit timer |
| | MSTP11 | Programmable pulse generator (PPG) |
| | MSTP10 | D/A converter |
| | MSTP9 | A/D converter |
| | MSTP8 | — |
| MSTPCRL | MSTP7 | Serial communication interface (SCI) channel 2 |
| | MSTP6 | Serial communication interface (SCI) channel 1 |
| | MSTP5 | Serial communication interface (SCI) channel 0 |
| | MSTP4 | — |
| | MSTP3 | — |
| | MSTP2 | — |
| | MSTP1 | — |
| | MSTP0 | — |

Note: Bit 8 and bits 4 to 0 can be read or written to, but do not affect operation.

### 21.5.2 Usage Notes

**DMAC/DTC Module Stop:** Depending on the operating status of the DMAC or DTC, the MSTP15 and MSTP14 bits may not be set to 1. Setting of the DMAC or DTC module stop mode should be carried out only when the respective module is not activated.

For details, refer to section 7, DMA Controller, and section 8, Data Transfer Controller.

**On-Chip Supporting Module Interrupt:** Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

**Writing to MSTPCR:** MSTPCR should only be written to by the CPU.

## 21.6 Software Standby Mode

### 21.6.1 Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip supporting modules, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip supporting modules other than the SCI, and I/O ports, are retained. Whether the address bus and bus control signals are placed in the high-impedance state or retain the output state can be specified by the OPE bit in SBYCR.

In this mode the oscillator stops, and therefore power dissipation is significantly reduced.

### 21.6.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins $\overline{IRQ_0}$ to $\overline{IRQ_2}$), or by means of the $\overline{RES}$ pin or $\overline{STBY}$ pin.

- Clearing with an interrupt

  When an NMI or $IRQ_0$ to $IRQ_2$ interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SYSCR, stable clocks are supplied to the entire H8S/2655 Series chip, software standby mode is cleared, and interrupt exception handling is started.

  When clearing software standby mode with an $IRQ_0$ to $IRQ_2$ interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts $IRQ_0$ to $IRQ_2$ is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side or has been designated as a DTC activation source.

- Clearing with the $\overline{RES}$ pin

  When the $\overline{RES}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire H8S/2655 Series chip. Note that the $\overline{RES}$ pin must be held low until clock oscillation stabilizes. When the $\overline{RES}$ pin goes high, the CPU begins reset exception handling.

- Clearing with the $\overline{STBY}$ pin

  When the $\overline{STBY}$ pin is driven low, a transition is made to hardware standby mode.

### 21.6.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode

Bits STS2 to STS0 in SBYCR should be set as described below.

**Using a Crystal Oscillator:** Set bits STS2 to STS0 so that the standby time is at least 8 ms (the oscillation stabilization time).

Table 21-4 shows the standby times for different operating frequencies and settings of bits STS2 to STS0.

**Table 21-4 Oscillation Stabilization Time Settings**

| STS2 | STS1 | STS0 | Standby Time | 20 MHz | 16 MHz | 12 MHz | 10 MHz | 8 MHz | 6 MHz | 4 MHz | 2 MHz | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 8192 states | 0.41 | 0.51 | 0.68 | 0.8 | 1.0 | 1.3 | 2.0 | 4.1 | ms |
| | | 1 | 16384 states | 0.82 | 1.0 | 1.3 | 1.6 | 2.0 | 2.7 | 4.1 | [8.2] | |
| | 1 | 0 | 32768 states | 1.6 | 2.0 | 2.7 | 3.3 | 4.1 | 5.5 | [8.2] | 16.4 | |
| | | 1 | 65536 states | 3.3 | 4.1 | 5.5 | 6.6 | [8.2] | [10.9] | 16.4 | 32.8 | |
| 1 | 0 | 0 | 131072 states | 6.6 | [8.2] | [10.9] | [13.1] | 16.4 | 21.8 | 32.8 | 65.5 | |
| | | 1 | 262144 states | [13.1] | 16.4 | 21.8 | 26.2 | 32.8 | 43.6 | 65.6 | 131.2 | |
| | 1 | 0 | —* | — | — | — | — | — | — | — | — | — |
| | | 1 | 16 states | 0.8 | 1.0 | 1.3 | 1.6 | 2.0 | 2.7 | 4.0 | 8.0 | µs |

[ ]: Recommended time setting *: Don't care

Note: * Reserved. If set, the standby time will be 16 states.

**Using an External Clock:** Any value can be set. Normally, use of the minimum time is recommended.

### 21.6.4 Software Standby Mode Application Example

Figure 21-2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in SYSCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

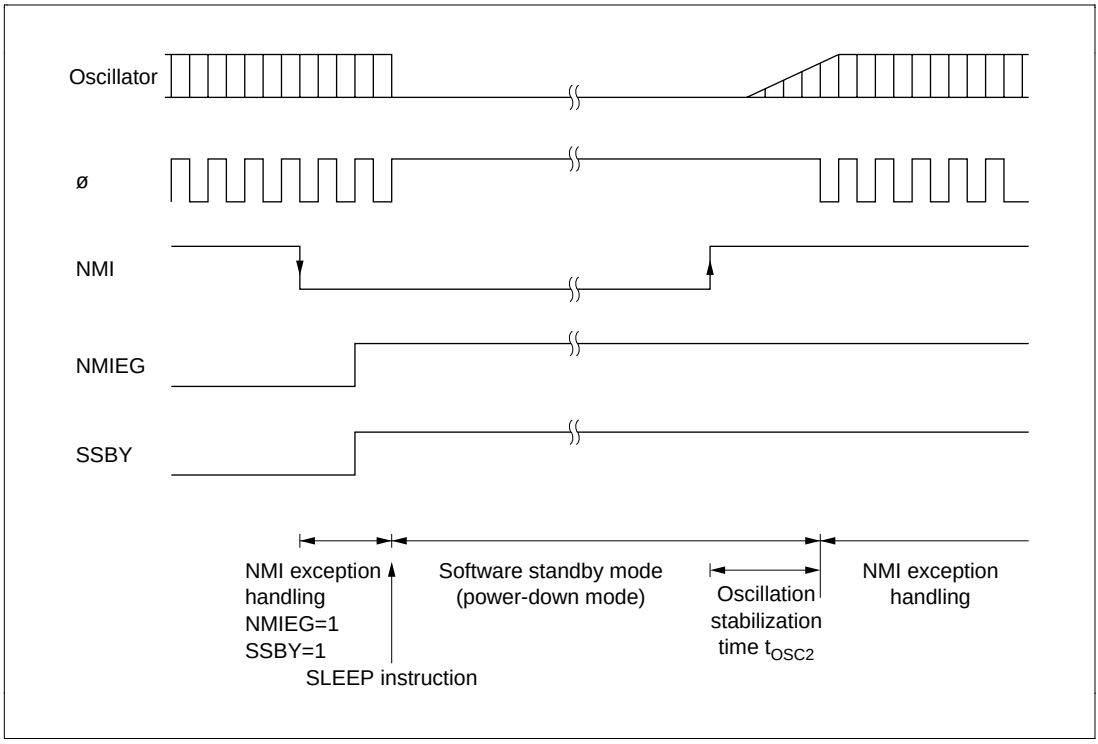



**Figure 21-2 Software Standby Mode Application Example**

### 21.6.5 Usage Notes

**I/O Port Status:** In software standby mode, I/O port states are retained. If the OPE bit is set to 1, the address bus and bus control signal output is also retained. Therefore, there is no reduction in current dissipation for the output current when a high-level signal is output.

**Current Dissipation during Oscillation Stabilization Wait Period:** Current dissipation increases during the oscillation stabilization wait period.

**Write Data Buffer Function:** The write data buffer function and software standby mode cannot be used at the same time. When the write data buffer function is used, the WDBE bit in BCRL should be cleared to 0 to cancel the write data buffer function before entering software standby mode. Also check that external writes have finished, by reading external addresses, etc., before executing a SLEEP instruction to enter software standby mode. See section 6.10, Write Data Buffer Function, for details of the write data buffer function.

## 21.7 Hardware Standby Mode

### 21.7.1 Hardware Standby Mode

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power dissipation. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the $\overline{\text{STBY}}$ pin low.

Do not change the state of the mode pins ($MD_2$ to $MD_0$) while the H8S/2655 Series is in hardware standby mode.

Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is set and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until the clock oscillator stabilizes (at least 8 ms—the oscillation stabilization time—when using a crystal oscillator). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

### 21.7.2 Hardware Standby Mode Timing

Figure 21-3 shows an example of hardware standby mode timing.

When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation stabilization time, then changing the $\overline{\text{RES}}$ pin from low to high.

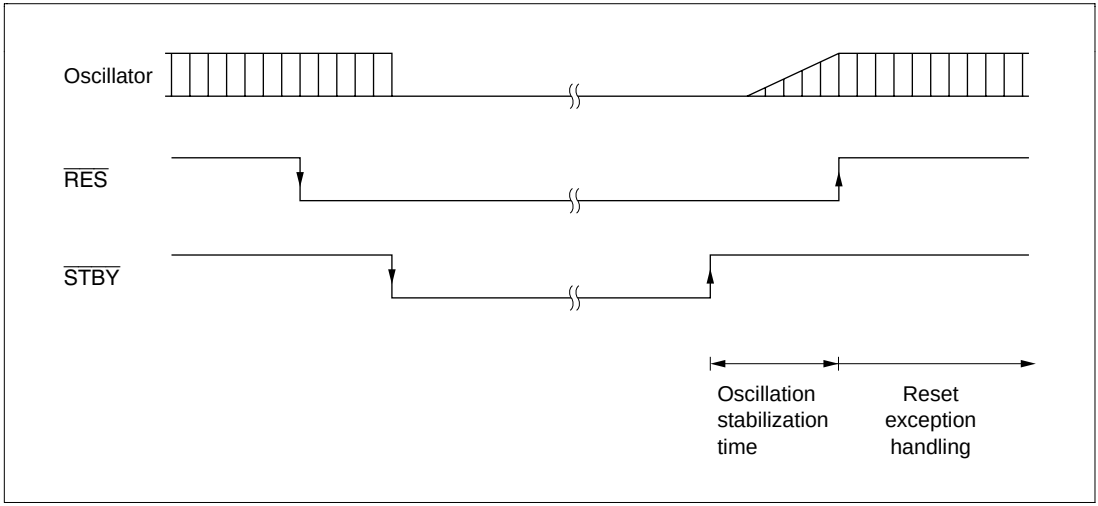



**Figure 21-3 Hardware Standby Mode Timing (Example)**

## 21.8 ø Clock Output Disabling Function

Output of the ø clock can be controlled by means of the PSTOP bit in SCKCR, and DDR for the corresponding port. When the PSTOP bit is set to 1, the ø clock stops at the end of the bus cycle, and ø output goes high. ø clock output is enabled when the PSTOP bit is cleared to 0. When DDR for the corresponding port is cleared to 0, ø clock output is disabled and input port mode is set. Table 21-5 shows the state of the ø pin in each processing state.

**Table 21-5 ø Pin State in Each Processing State**

| DDR | 0 | 1 | |
|---|---|---|---|
| PSTOP | — | 0 | 1 |
| Hardware standby mode | High impedance | | |
| Software standby mode | High impedance | Fixed high | |
| Sleep mode | High impedance | ø output | Fixed high |
| Normal operating state | High impedance | ø output | Fixed high |

# Section 22 Electrical Characteristics

## 22.1 Absolute Maximum Ratings

Table 22-1 lists the absolute maximum ratings.

**Table 22-1 Absolute Maximum Ratings**

| Item | Symbol | Value | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$ | –0.3 to +7.0 | V |
| Programming voltage | $V_{PP}$ | –0.3 to +13.5 | V |
| Input voltage (except port 4) | $V_{in}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Input voltage (port 4) | $V_{in}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Reference voltage | $V_{ref}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Analog power supply voltage | $AV_{CC}$ | –0.3 to +7.0 | V |
| Analog input voltage | $V_{AN}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Operating temperature | $T_{opr}$ | Regular specifications: –20 to +75 | °C |
| | | Wide-range specifications: –40 to +85 | °C |
| Storage temperature | $T_{stg}$ | –55 to +125 | °C |

Caution: Permanent damage to the chip may result if absolute maximum rating are exceeded.

## 22.2 DC Characteristics

Table 22-2 lists the DC characteristics. Table 22-3 lists the permissible output currents.

**Table 22-2 DC Characteristics**

Conditions: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $P6_4$ to $P6_7$, $PA_4$ to $PA_7$ | $V_T^-$ | 1.0 | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | 0.4 | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | $V_{IH}$ | $V_{CC} - 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | Port 1, 3, 5, B to G, $P6_0$ to $P6_3$, $PA_0$ to $PA_3$ | | 2.0 | — | $V_{CC} + 0.3$ | V | |
| | Port4 | | 2.0 | — | $AV_{CC} + 0.3$ | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$ | $V_{IL}$ | –0.3 | — | 0.5 | V | |
| | NMI, EXTAL, Port 1, 3 to 5, B to G, $P6_0$ to $P6_3$, $PA_0$ to $PA_3$ | | –0.3 | — | 0.8 | V | |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC} - 0.5$ | — | — | V | $I_{OH}$ = –200 µA |
| | | | 3.5 | — | — | V | $I_{OH}$ = –1 mA |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | Port 1, A to C | | — | — | 1.0 | V | $I_{OL}$ = 10 mA |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC}$ – 0.5 V |

Note: 1. If the A/D and D/A converters are not used,do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

**Table 22-2 DC Characteristics (cont)**

Conditions: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, 5, 6, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 50 | — | 300 | µA | $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in}$ = 0 V, f = 1 Mhz, $T_a$ = 25°C |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | 80 (5.0 V) | 122 | mA | f = 20 MHz |
| | Sleep mode | | — | 60 (5.0 V) | 84 | mA | f = 20 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \le 50°C$ |
| | | | — | — | 20 | | 50°C < $T_a$ |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | 16 (5.0V) | 24 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | 2 | 3.0 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used,do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.
2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.
3. The values are for $V_{RAM} \le V_{CC} < 4.5V$, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3 V.
4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
$I_{CC}$ max = 1.0 (mA) + 1.1 (mA/(MHz × V)) × $V_{CC}$ × f [normal mode]
$I_{CC}$ max = 1.0 (mA) + 0.75 (mA/(MHz × V)) × $V_{CC}$ × f [sleep mode]

**Table 22-2 DC Characteristics (cont)**

Conditions: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $P6_4$ to $P6_7$, $PA_4$ to $PA_7$ | $V_T^-$ | $V_{CC} \times 0.2$ | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | $V_{CC} \times 0.07$ | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | $V_{IH}$ | $V_{CC} \times 0.9$ | — | $V_{CC}$ +0.3 | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 1, 3, 5, B to G, $P6_0$ to $P6_3$, $PA_0$ to $PA_3$ | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 4 | | $V_{CC} \times 0.7$ | — | $AV_{CC}$ +0.3 | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$ | $V_{IL}$ | –0.3 | — | $V_{CC} \times 0.1$ | V | |
| | NMI, EXTAL, Port 1, 3 to 5, B to G, $P6_0$ to $P6_3$, $PA_0$ to $PA_3$ | | –0.3 | — | $V_{CC} \times 0.2$ | V | $V_{CC}$ < 4.0 V |
| | | | | | 0.8 | | $V_{CC}$ = 4.0 to 5.5 V |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | $V_{CC}$ – 1.0 | — | — | V | $I_{OH}$ = –1 mA |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | Port 1, A to C | | — | — | 1.0 | V | $V_{CC} \leq 4$ V<br>$I_{OL}$ = 5 mA<br>4.0 < $V_{CC} \leq$ 5.5 V<br>$I_{OL}$ = 10 mA |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC}$ – 0.5V |

Note: 1. If the A/D and D/A converters are not used,do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

**Table 22-2 DC Characteristics (cont)**

Conditions: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, 5, 6, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ –0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 10 | — | 300 | µA | $V_{CC}$ = 2.7 V to 5.5 V, $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in}$ = 0 V<br>f = 1 Mhz<br>Ta = 25°C |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | 25 (3.0 V) | 62 | mA | f = 10 MHz |
| | Sleep mode | | — | 18 (3.0 V) | 42 | mA | f = 10 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \le 50°C$ |
| | | | — | — | 20 | | 50°C < $T_a$ |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | 12 (3.0V) | 22 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | 1.5 (3.0 V) | 2.5 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used,do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.

3. The values are for $V_{RAM} \le V_{CC}$ < 2.7 V, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3V.

4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
$I_{CC}$ max = 1.0 (mA) + 1.1 (mA/(MHz × V)) × $V_{CC}$ × f [normal mode]
$I_{CC}$ max = 1.0 (mA) + 0.75 (mA/(MHz × V)) × $V_{CC}$ × f [sleep mode]

**Table 22-3 Permissible Output Currents**

Conditions: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Permissible output low current (per pin) | Port 1, A to C | $I_{OL}$ | — | — | 10 | mA |
| | Other output pins | | — | — | 2.0 | mA |
| Permissible output low current (total) | Total of 32 pins including port 1 and A to C | $\sum I_{OL}$ | — | — | 80 | mA |
| | Total of all output pins, including the above | | — | — | 120 | mA |
| Permissible output high current (per pin) | All output pins | $-I_{OH}$ | — | — | 2.0 | mA |
| Permissible output high current (total) | Total of all output pins | $\sum -I_{OH}$ | — | — | 40 | mA |

Notes: 1. To protect chip reliability, do not exceed the output current values in table 22-3.
2. When driving a darlington pair or LED directly, always insert a current-limiting resistor in the output line, as show in figures 22-1 and 22-2.

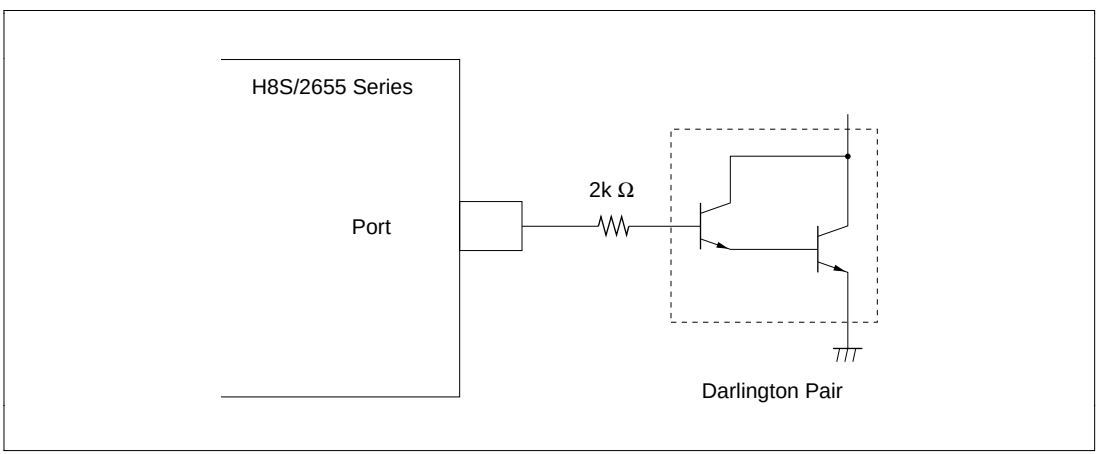



**Figure 22-1 Darlington Pair Drive Circuit (Example)**

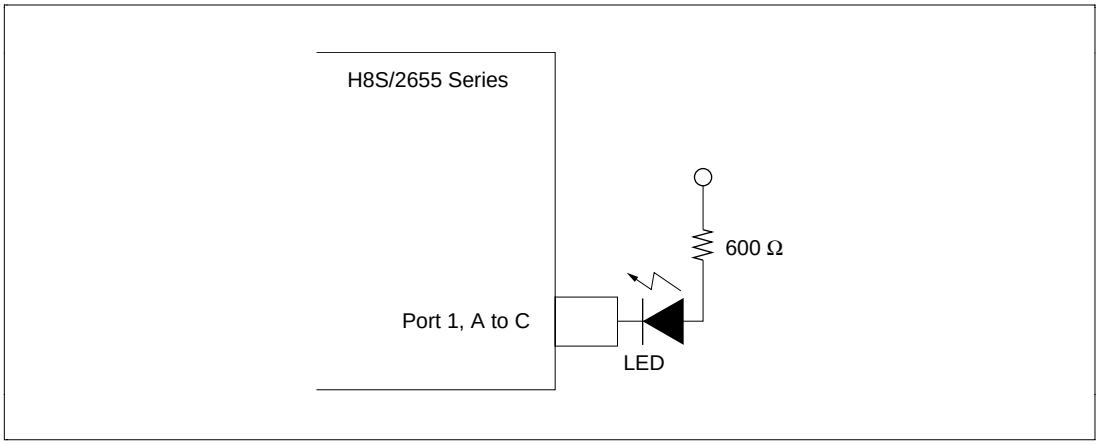



**Figure 22-2 LED Drive Circuit (Example)**

## 22.3 AC Characteristics

Figure 22-3 show, the test conditions for the AC characteristics.

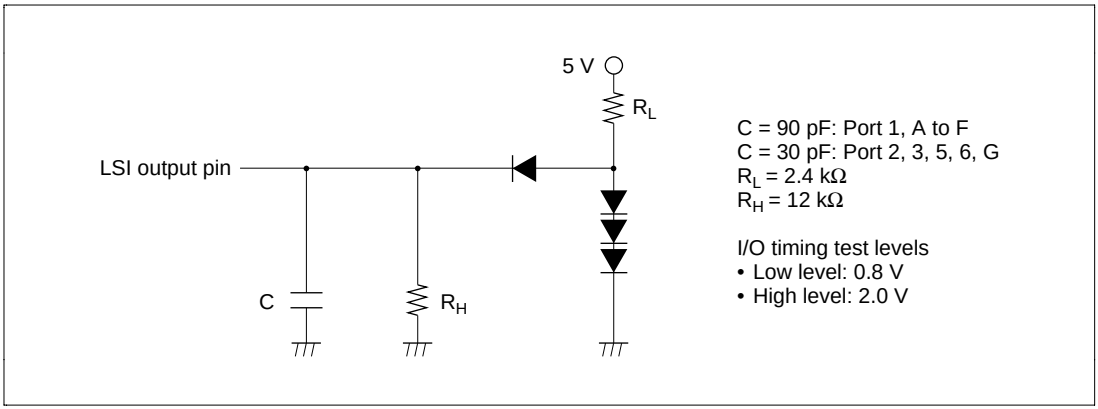



**Figure 22-3 Output Load Circuit**

### 22.3.1 Clock Timing

Table 22-4 lists the clock timing

**Table 22-4 Clock Timing**

Condition A: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Clock cycle time | $t_{cyc}$ | 100 | 500 | 50 | 500 | ns | Figure 22-4 |
| Clock high pulse width | $t_{CH}$ | 35 | — | 20 | — | ns | Figure 22-4 |
| Clock low pulse width | $t_{CL}$ | 35 | — | 20 | — | ns | |
| Clock rise time | $t_{Cr}$ | — | 15 | — | 5 | ns | |
| Clock fall time | $t_{Cf}$ | — | 15 | — | 5 | ns | |
| Clock oscillator setting time at reset (crystal) | $t_{OSC1}$ | 20 | — | 10 | — | ms | Figure 22-5 |
| Clock oscillator setting time in software standby (crystal) | $t_{OSC2}$ | 20 | — | 10 | — | ms | Figure 21-2 |
| External clock output stabilization delay time | $t_{DEXT}$ | 500 | — | 500 | — | µs | Figure 22-5 |

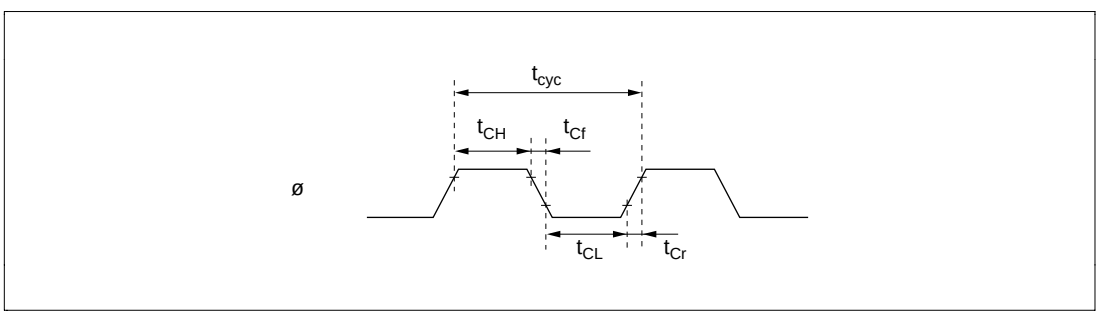



**Figure 22-4 System Clock Timing**

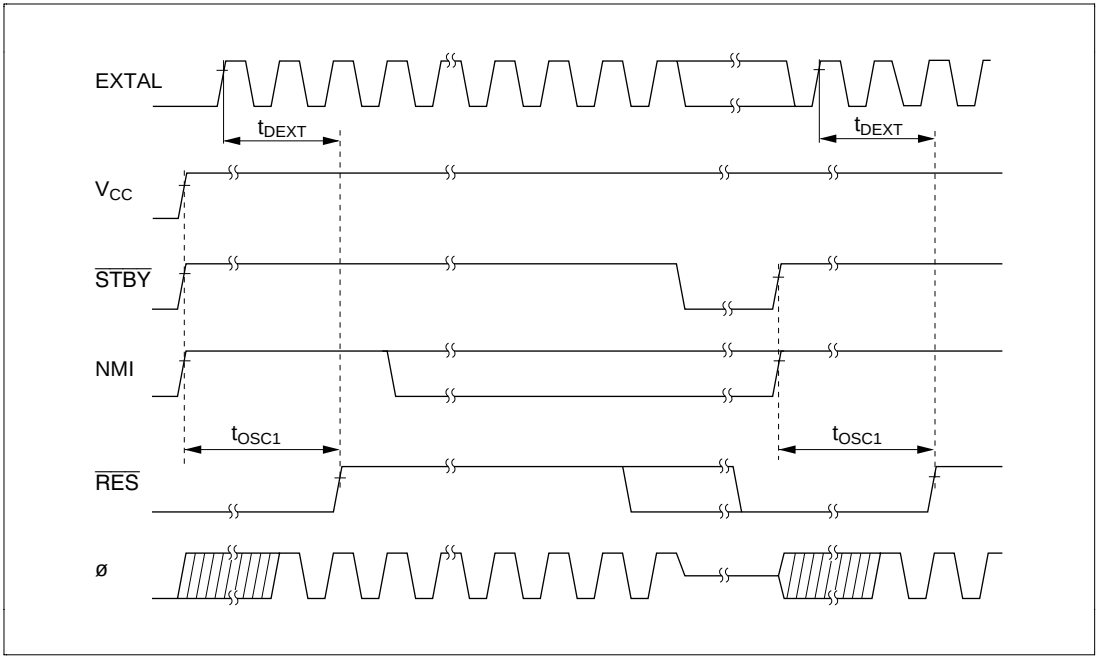



**Figure 22-5 Oscillator Settling Timing**

### 22.3.2 Control Signal Timing

Table 22-5 lists the control signal timing.

**Table 22-5 Control Signal Timing**

Condition A: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{RES}$ setup time | $t_{RESS}$ | 200 | — | 200 | — | ns | Figure 22-6 |
| $\overline{RES}$ pulse width | $t_{RESW}$ | 20 | — | 20 | — | $t_{cyc}$ | |
| NMI reset setup time | $t_{NMIRS}$ | 250 | — | 200 | — | ns | |
| NMI reset hold time | $t_{NMIRH}$ | 200 | — | 200 | — | | |
| NMI setup time | $t_{NMIS}$ | 250 | — | 150 | — | ns | Figure 22-7 |
| NMI hold time | $t_{NMIH}$ | 10 | — | 10 | — | | |
| NMI pulse width (exiting software standby mode) | $t_{NMIW}$ | 200 | — | 200 | — | ns | |
| $\overline{IRQ}$ setup time | $t_{IRQS}$ | 250 | — | 150 | — | ns | |
| $\overline{IRQ}$ hold time | $t_{IRQH}$ | 10 | — | 10 | — | ns | |
| $\overline{IRQ}$ pulse width (exiting software standby mode) | $t_{IRQW}$ | 200 | — | 200 | — | ns | |

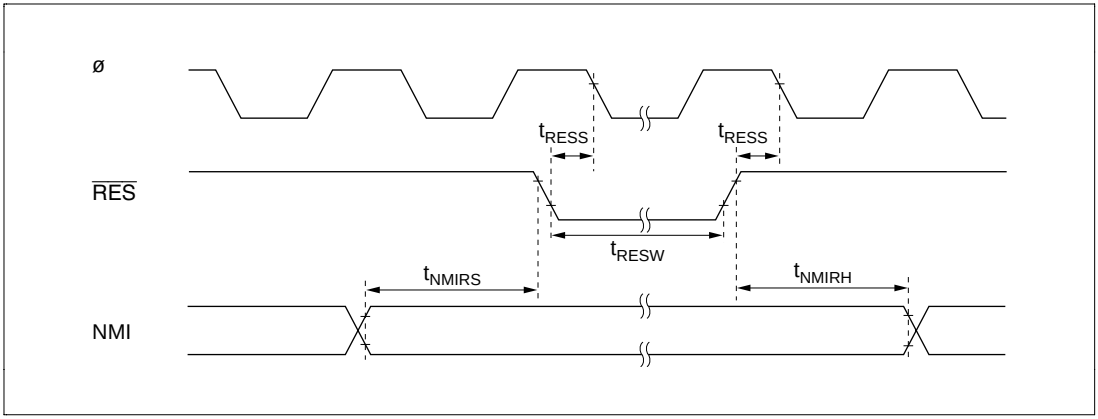



**Figure 22-6 Reset Input Timing**

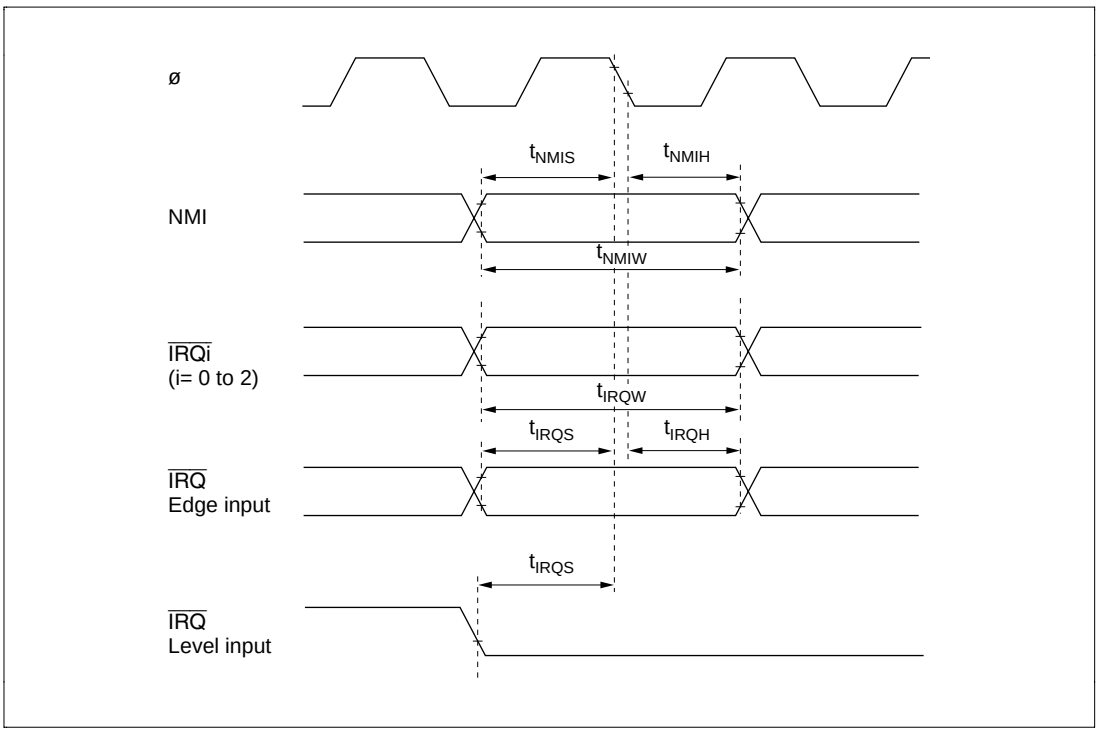



**Figure 22-7 Interrupt Input Timing**

### 22.3.3 Bus Timing

Table 22-6 lists the bus timing.

**Table 22-6 Bus Timing**

Condition A: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø= 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Address delay time | $t_{AD}$ | — | 40 | — | 20 | ns | Figure 22-8 to Figure 22-18 |
| Address setup time | $t_{AS}$ | 0.5 × $t_{cyc}$ – 30 | — | 0.5 × $t_{cyc}$ – 15 | — | ns | |
| Address hold time | $t_{AH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| Precharge time | $t_{PCH}$ | 1.5 × $t_{cyc}$ – 40 | — | 1.5 × $t_{cyc}$ – 20 | — | ns | |
| $\overline{CS}$ delay time 1 | $t_{CSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{CS}$ delay time 2 | $t_{CSD2}$ | — | 40 | — | 20 | ns | |
| $\overline{CS}$ pulse width | $t_{CSW}$ | 2.5 × $t_{cyc}$ – 40 | — | 2.5 × $t_{cyc}$ – 20 | — | ns | |
| $\overline{AS}$ delay time | $t_{ASD}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 1 | $t_{RSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 2 | $t_{RSD2}$ | — | 40 | — | 20 | ns | |
| $\overline{CAS}$ delay time | $t_{CASD}$ | — | 40 | — | 20 | ns | |
| Read data setup time | $t_{RDS}$ | 30 | — | 15 | — | ns | |
| Read data hold time | $t_{RDH}$ | 0 | — | 0 | — | ns | |
| Read data access time1 | $t_{ACC1}$ | — | 1.0 × $t_{cyc}$ – 50 | — | 1.0 × $t_{cyc}$ – 25 | ns | |
| Read data access time2 | $t_{ACC2}$ | — | 1.5 × $t_{cyc}$ – 50 | — | 1.5 × $t_{cyc}$ – 25 | ns | |
| Read data access time3 | $t_{ACC3}$ | — | 2.0 × $t_{cyc}$ – 50 | — | 2.0 × $t_{cyc}$ – 25 | ns | |

**Table 22-6 Bus Timing (cont)**

Condition A: $V_{CC}$ = 2.7 to 5.5 V, $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, ø= 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Read data access time 4 | $t_{ACC4}$ | — | 2.5 × $t_{cyc}$ – 50 | — | 2.5 × $t_{cyc}$ – 25 | ns | Figure 22-8 to Figure 22-18 |
| Read data access time 5 | $t_{ACC5}$ | — | 3.0 × $t_{cyc}$ – 50 | — | 3.0 × $t_{cyc}$ – 25 | ns | |
| $\overline{WR}$ delay time 1 | $t_{WRD1}$ | — | 40 | — | 20 | ns | |
| $\overline{WR}$ delay time 2 | $t_{WRD2}$ | — | 40 | — | 20 | ns | |
| $\overline{WR}$ pulse width 1 | $t_{WSW1}$ | 1.0 × $t_{cyc}$ – 40 | — | 1.0 × $t_{cyc}$ – 20 | — | ns | |
| $\overline{WR}$ pulse width 2 | $t_{WSW2}$ | 1.5 × $t_{cyc}$ – 40 | — | 1.5 × $t_{cyc}$ – 20 | — | ns | |
| Write data delay time | $t_{WDD}$ | — | 60 | — | 30 | ns | |
| Write data setup time | $t_{WDS}$ | 0.5 × $t_{cyc}$ – 40 | — | 0.5 × $t_{cyc}$ – 20 | — | ns | |
| Write data hold time | $t_{WDH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{WR}$ setup time | $t_{WCS}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{WR}$ hold time | $t_{WCH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{CAS}$ setup time | $t_{CSR}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | Figure 22-12 |
| $\overline{WAIT}$ setup time | $t_{WTS}$ | 60 | — | 30 | — | ns | Figure 22-10 |
| $\overline{WAIT}$ hold time | $t_{WTH}$ | 10 | — | 5 | — | ns | |
| $\overline{BREQ}$ setup time | $t_{BRQS}$ | 60 | — | 30 | — | ns | Figure 22-19 |
| $\overline{BACK}$ delay time | $t_{BACD}$ | — | 30 | — | 15 | ns | |
| Bus-floating time | $t_{BZD}$ | — | 100 | — | 50 | ns | |
| $\overline{BREQO}$ delay time | $t_{BRQOD}$ | — | 60 | — | 30 | ns | Figure 22-20 |

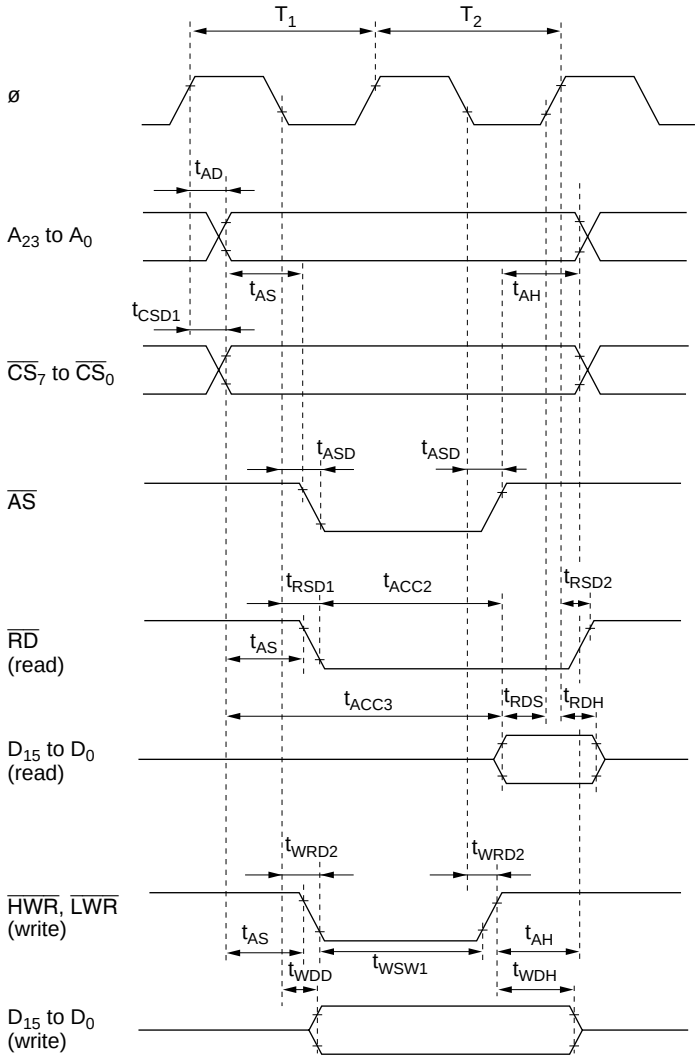



**Figure 22-8 Basic Bus Timing (Two-State Access)**

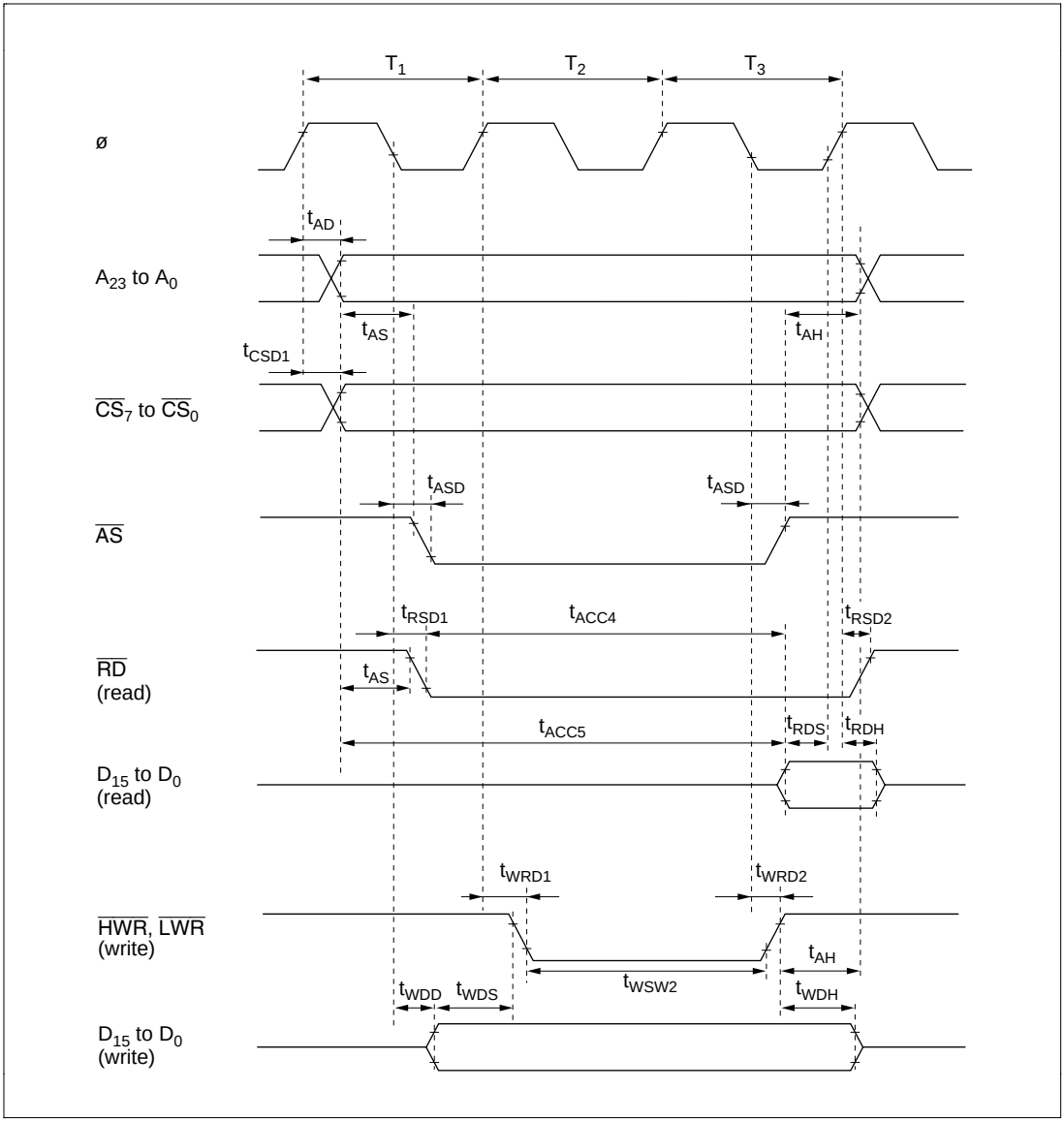



**Figure 22-9 Basic Bus Timing (Three-State Access)**

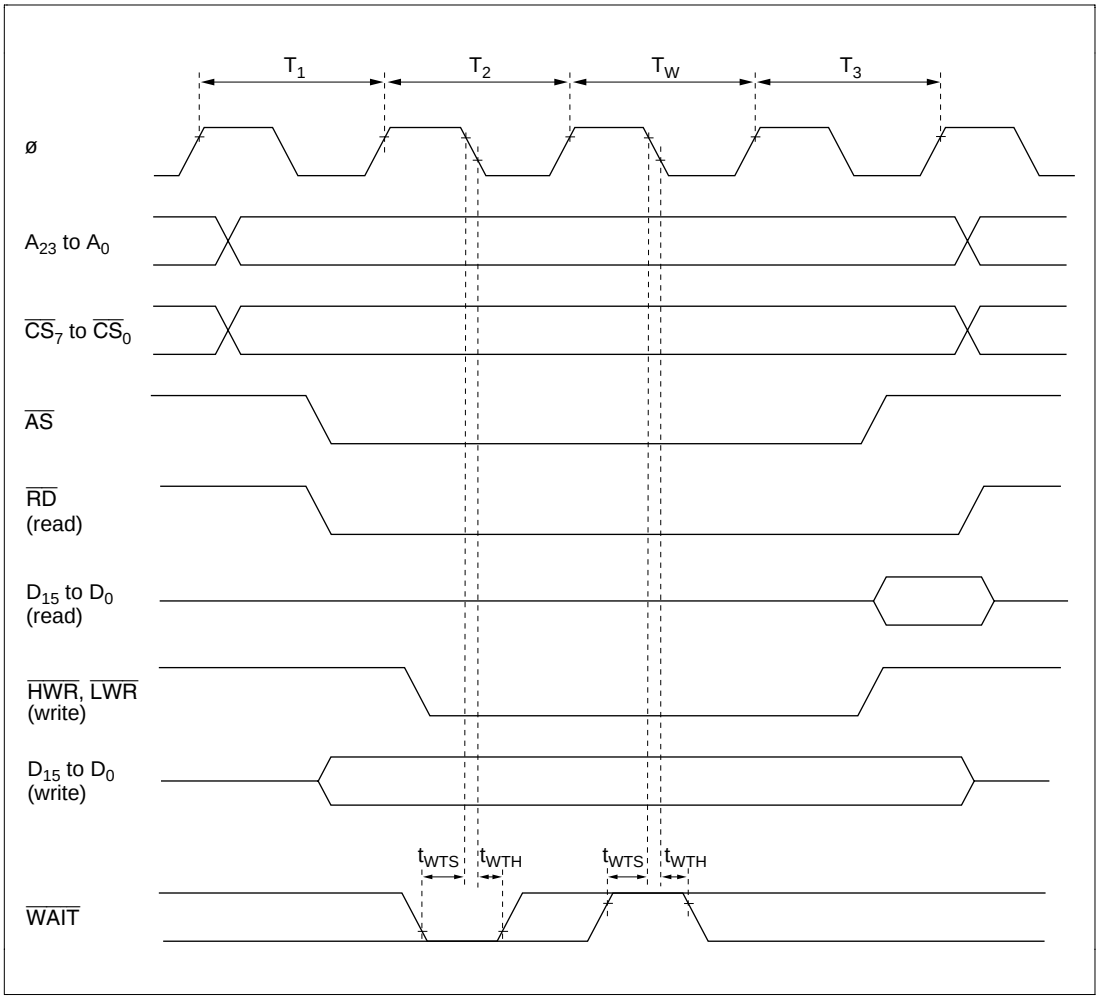



**Figure 22-10 Basic Bus Timing (Three-State Access with One Wait State)**

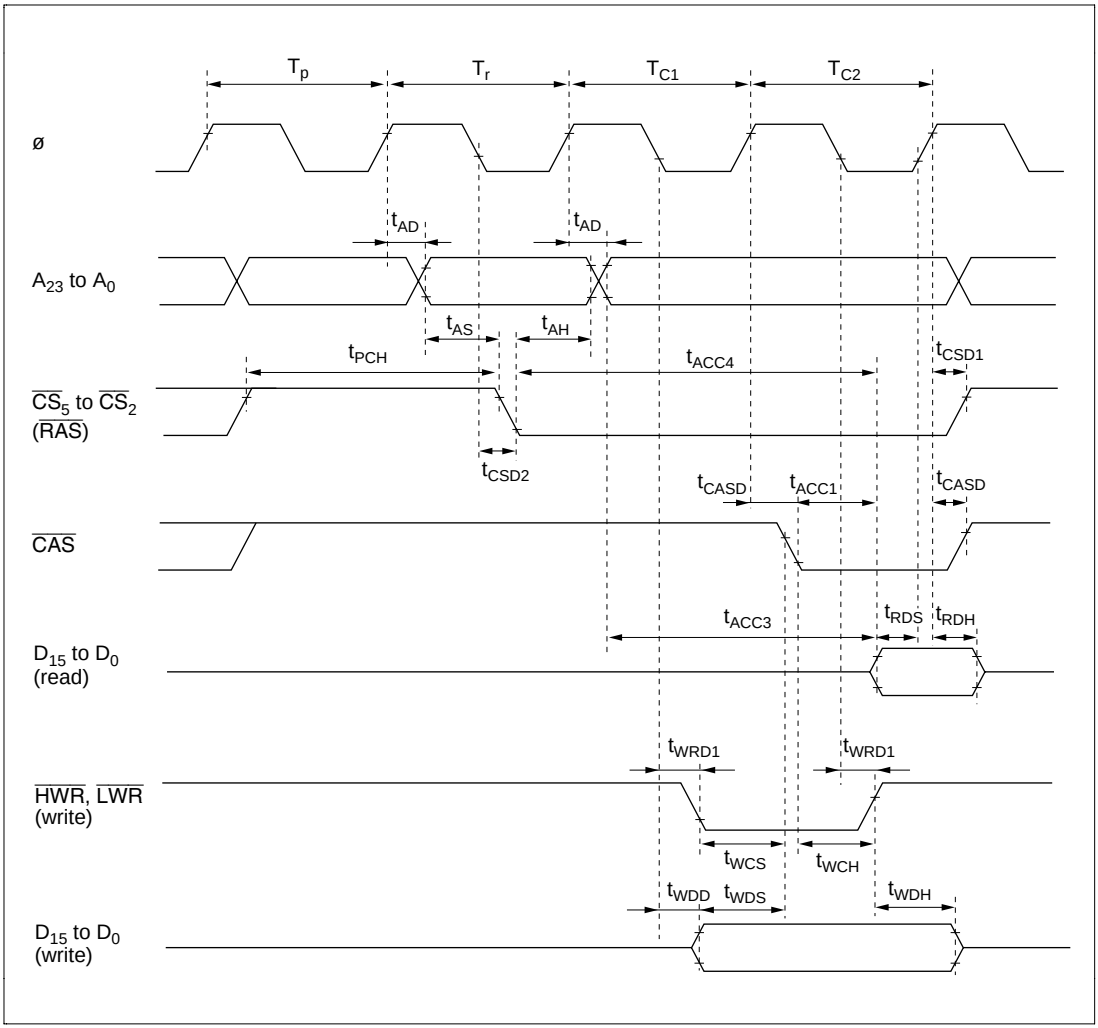



**Figure 22-11 DRAM Bus Timing**

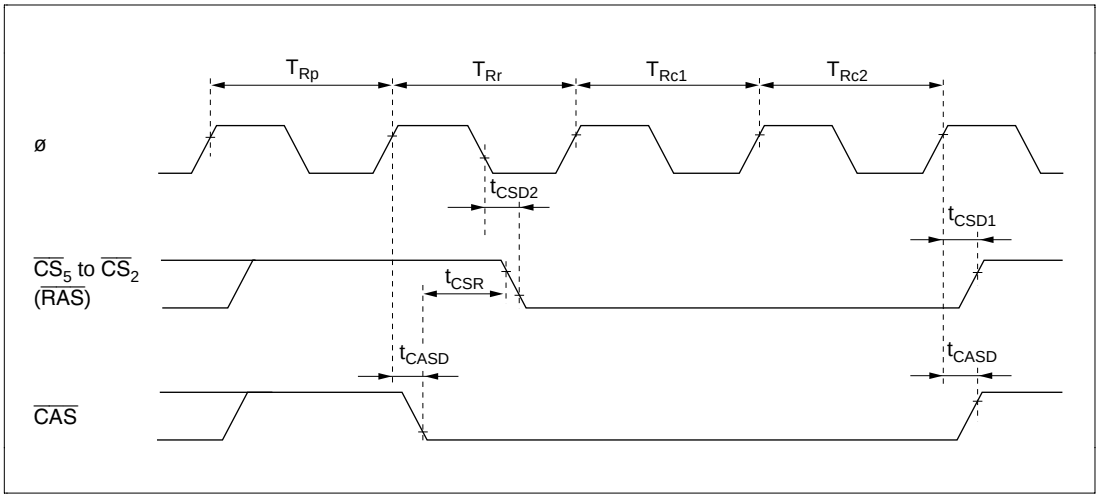



**Figure 22-12 CAS-Before-RAS Refresh Timing**

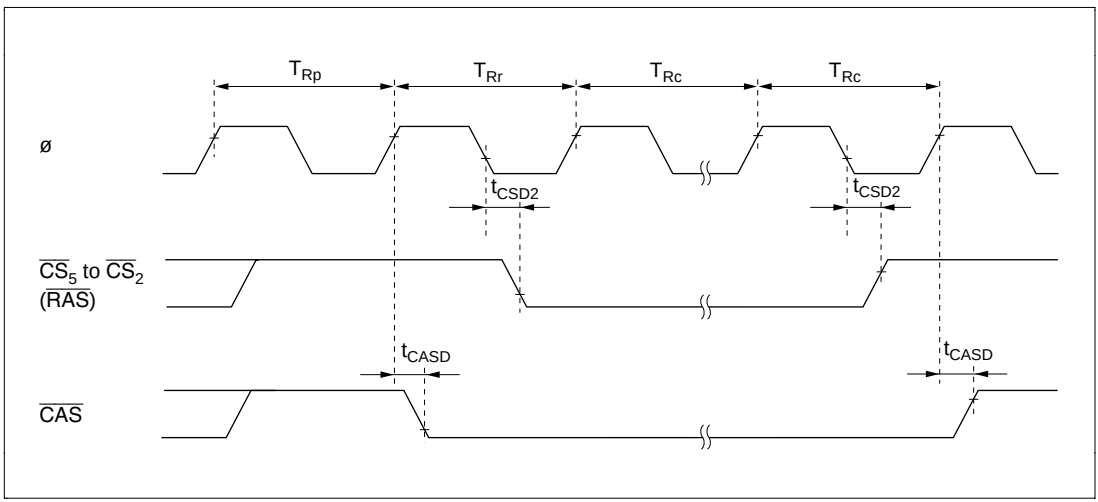



**Figure 22-13 Self-Refresh Timing**

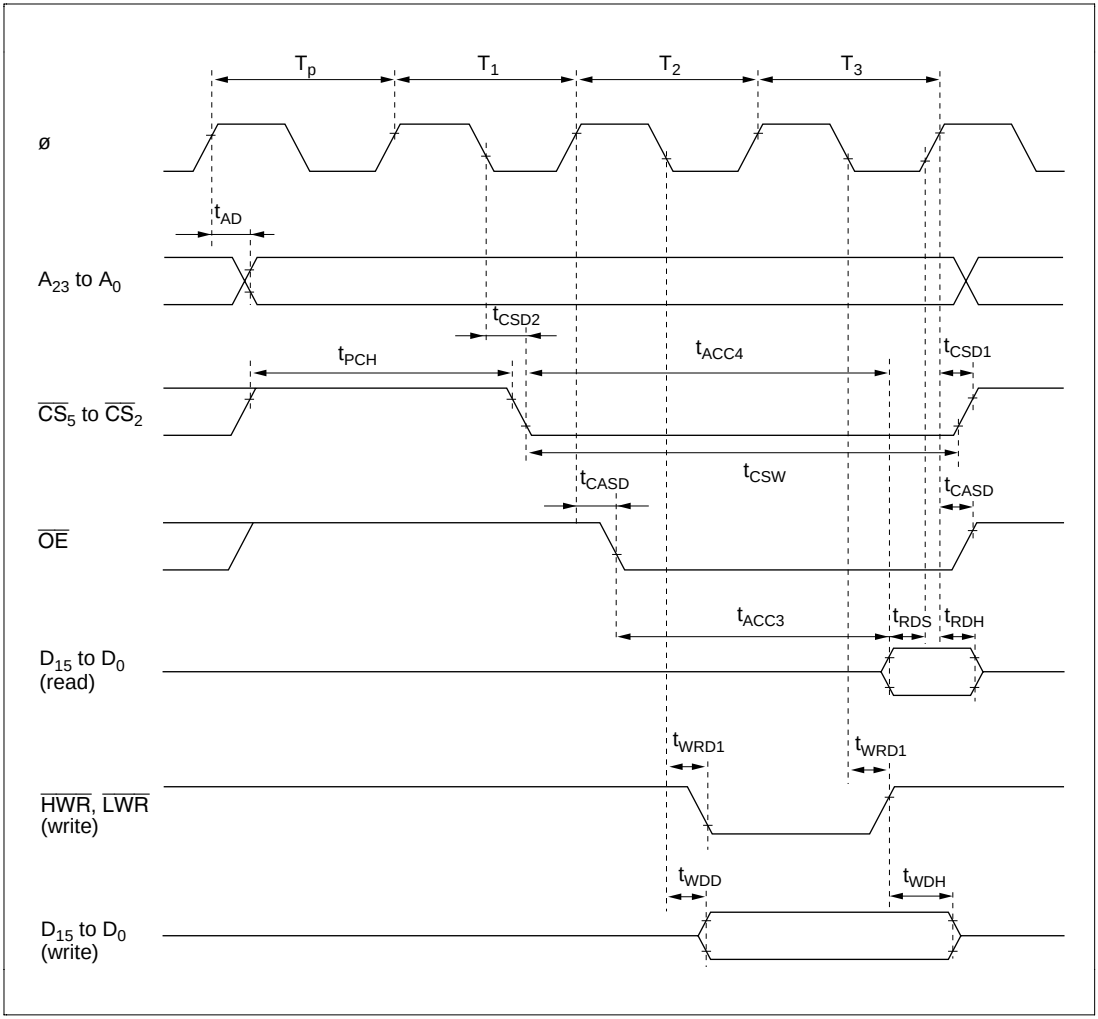



**Figure 22-14 PSRAM Bus Timing**

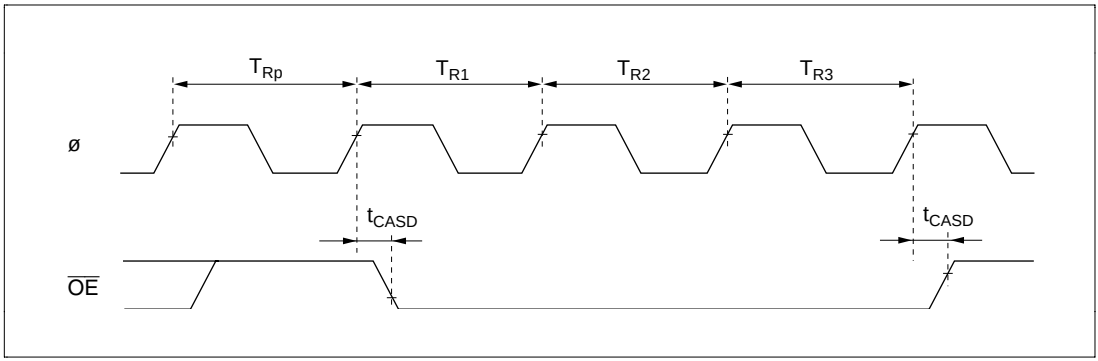



**Figure 22-15 Auto Refresh Timing**

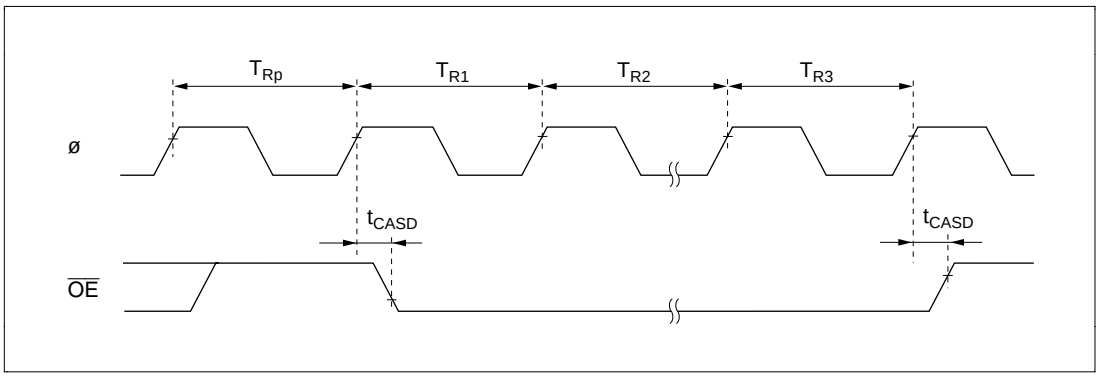



**Figure 22-16 Self-Refresh Timing**

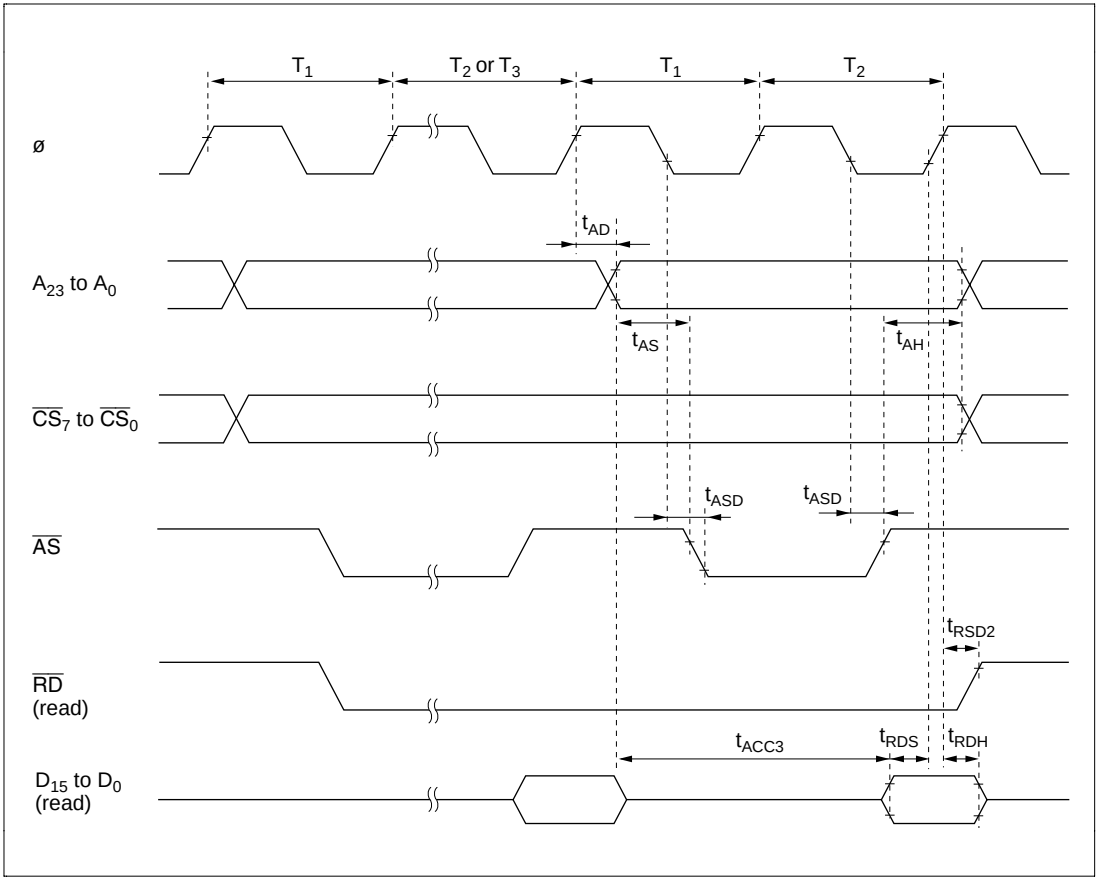



**Figure 22-17 Burst ROM Access Timing (Two-State Access)**

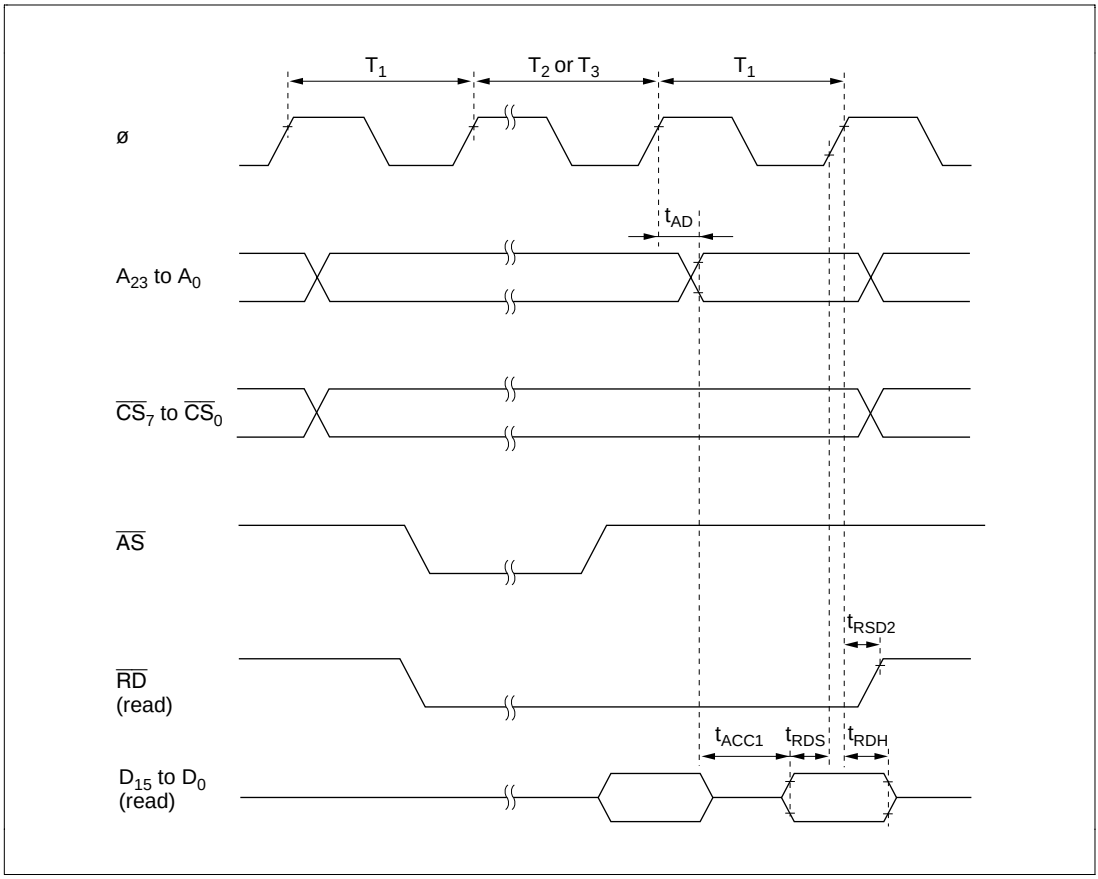



**Figure 22-18 Burst ROM Access Timing (One-State Access)**

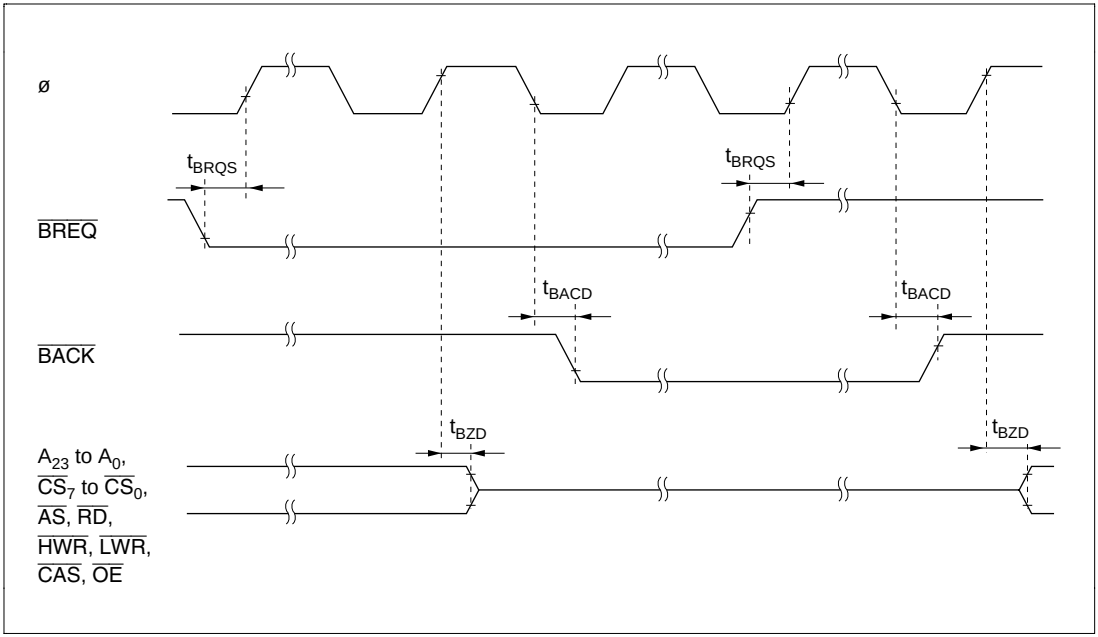



**Figure 22-19 External Bus Release Timing**

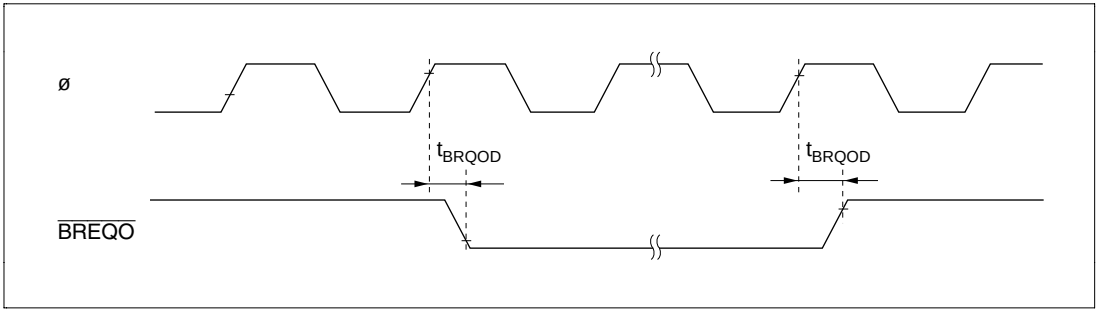



**Figure 22-20 External Bus Request Output Timing**

### 22.3.4 DMAC Timing

Table 22-7 lists the DMAC timing.

**Table 22-7 DMAC Timing**

Condition A: $V_{CC}$ = 2.7 V to 5.5 V, $AV_{CC}$ = 2.7 V to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{\text{DREQ}}$ setup time | $t_{DRQS}$ | 40 | — | 30 | — | ns | Figure 22-24 |
| $\overline{\text{DREQ}}$ hold time | $t_{DRQH}$ | 10 | — | 10 | — | | |
| $\overline{\text{TEND}}$ delay time | $t_{TED}$ | — | 40 | — | 20 | | Figure 22-23 |
| $\overline{\text{DACK}}$ delay time 1 | $t_{DACD1}$ | — | 40 | — | 20 | ns | Figure 22-21, Figure 22-22 |
| $\overline{\text{DACK}}$ delay time 2 | $t_{DACD2}$ | — | 40 | — | 20 | | |

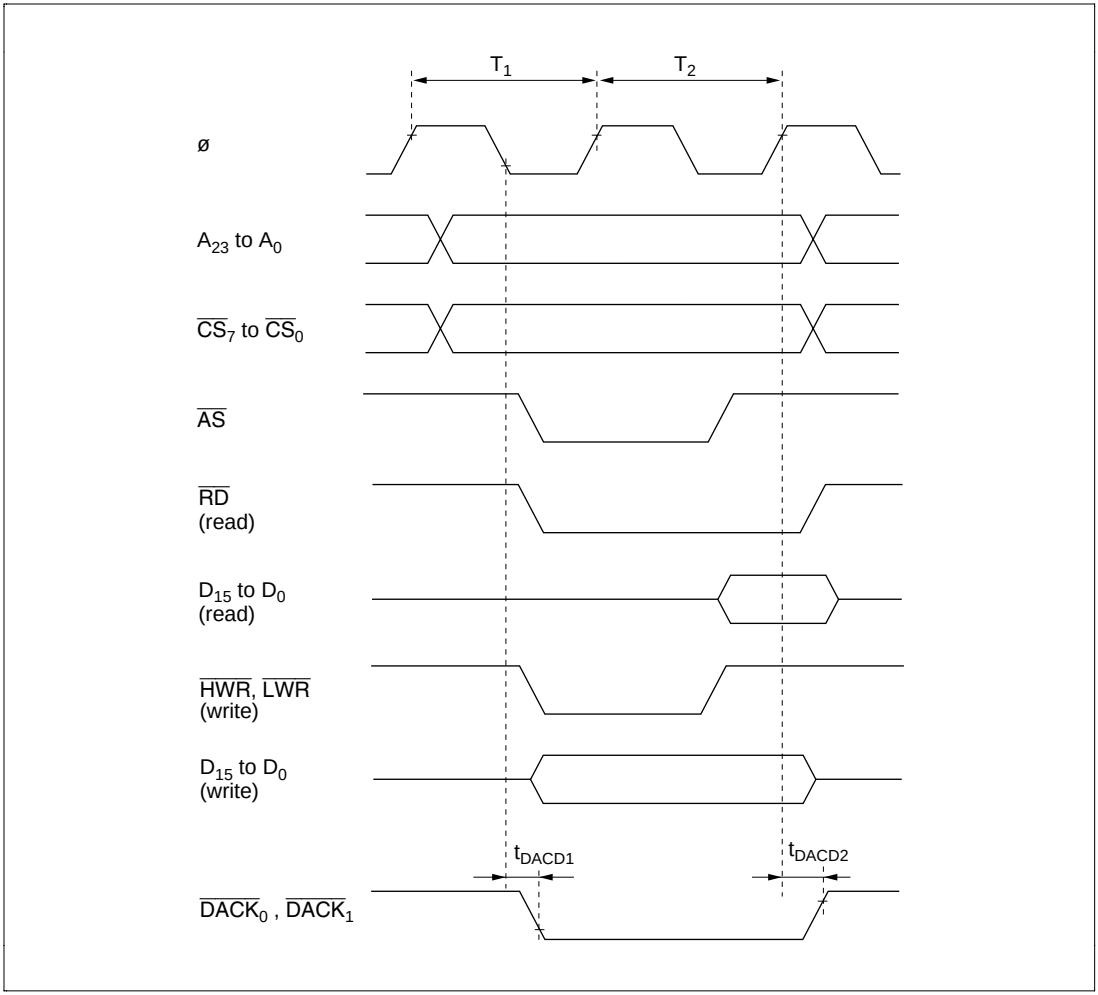



**Figure 22-21 DMAC Single Address Transfer Timing (Two-State Access)**

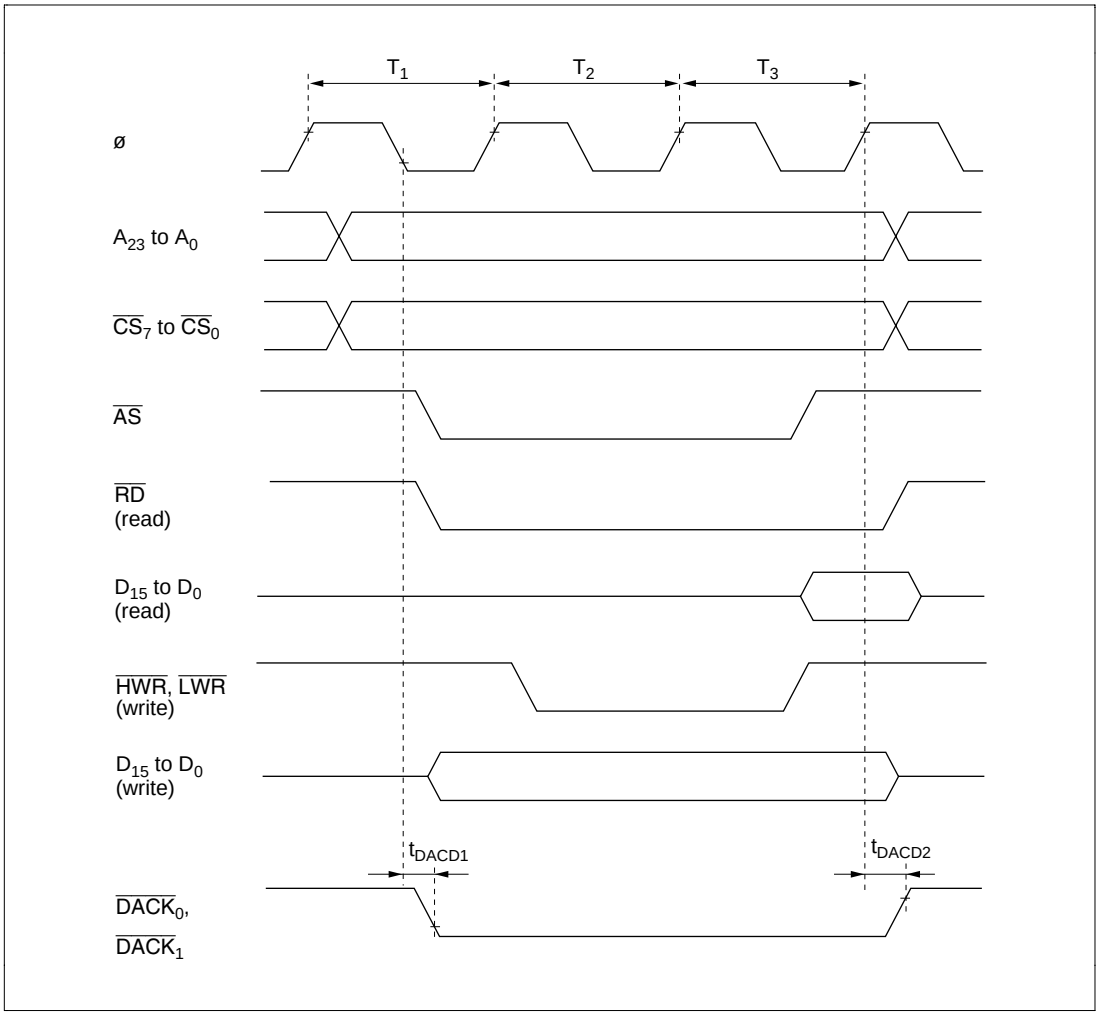



**Figure 22-22 DMAC Single Address Transfer Timing (Three-State Access)**

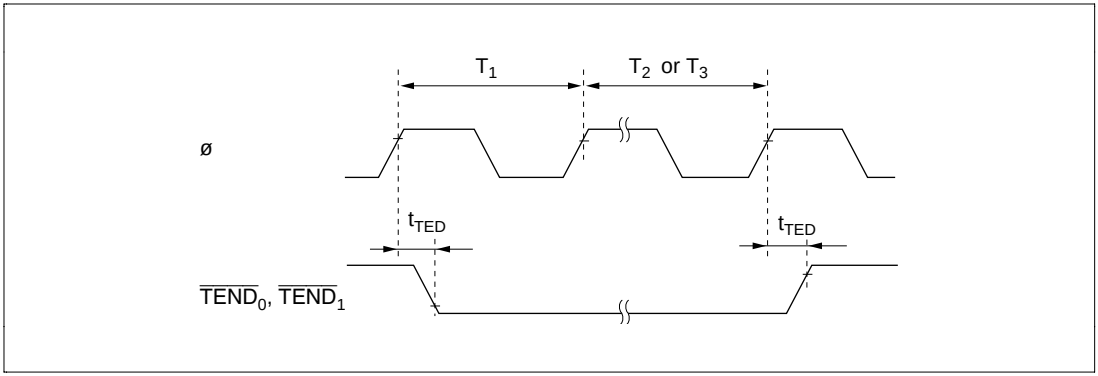



**Figure 22-23 DMAC $\overline{\text{TEND}}$ Output Timing**

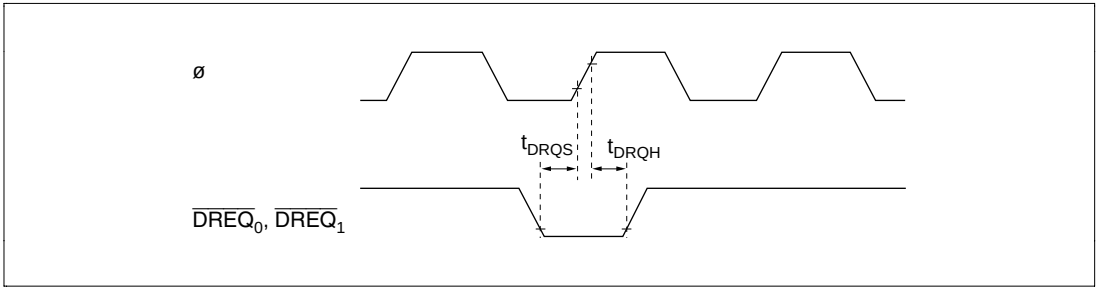



**Figure 22-24 DMAC $\overline{\text{DREQ}}$ Intput Timing**

### 22.3.5 Timing of On-Chip Supporting Modules

Table 22-8 lists the timing of on-chip supporting modules.

**Table 22-8 Timing of On-Chip Supporting Modules**

Condition A: $V_{CC}$ = 2.7 V to 5.5 V, $AV_{CC}$ = 2.7 V to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$,
$V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| I/O PORTS | Output data delay time | | $t_{PWD}$ | — | 100 | — | 50 | ns | Figure 22-25 |
| | Input data setup time | | $t_{PRS}$ | 50 | — | 30 | — | | |
| | Input data hold time | | $t_{PRH}$ | 50 | — | 30 | — | | |
| PPG | Pulse output delay time | | $t_{POD}$ | — | 100 | — | 50 | ns | Figure 22-26 |
| TPU | Timer output delay time | | $t_{TOCD}$ | — | 100 | — | 50 | ns | Figure 22-27 |
| | Timer input setup time | | $t_{TICS}$ | 50 | — | 30 | — | | |
| | Timer clock input setup time | | $t_{TCKS}$ | 50 | — | 30 | — | ns | Figure 22-28 |
| | Timer clock pulse width | Single edge | $t_{TCKWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | | Both edges | $t_{TCKWL}$ | 2.5 | — | 2.5 | — | | |
| TMR | Timer output delay time | | $t_{TMOD}$ | — | 100 | — | 50 | ns | Figure 22-29 |
| | Timer reset input setup time | | $t_{TMRS}$ | 50 | — | 30 | — | ns | Figure 22-31 |
| | Timer clock input setup time | | $t_{TMCS}$ | 50 | — | 30 | — | ns | Figure 22-30 |

**Table 22-8 Timing of On-Chip Supporting Modules (cont)**

Condition A: $V_{CC}$ = 2.7 V to 5.5 V, $AV_{CC}$ = 2.7 V to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = 5.0 V ± 10%, $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| TMR | Timer clock pulse width | Single edge | $t_{TMCWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | Figure 22-30 |
| | | Both edges | $t_{TMCWL}$ | 2.5 | — | 2.5 | — | | |
| WDT | Overflow output delay time | | $t_{WOVD}$ | — | 100 | — | 50 | ns | Figure 22-32 |
| SCI | Input clock cycle | Asynchro-nous | $t_{Scyc}$ | 4 | — | 4 | — | $t_{cyc}$ | Figure 22-33 |
| | | Synchro-nous | | 6 | — | 6 | — | | |
| | Input clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Input clock rise time | | $t_{SCKr}$ | — | 1.5 | — | 1.5 | $t_{cyc}$ | |
| | Input clock fall time | | $t_{SCKf}$ | — | 1.5 | — | 1.5 | | |
| | Transmit data delay time | | $t_{TXD}$ | — | 100 | — | 50 | ns | Figure 22-34 |
| | Receive data setup time (synchronous) | | $t_{RXS}$ | 100 | — | 50 | — | ns | |
| | Receive data hold time (synchronous) | | $t_{RXH}$ | 100 | — | 50 | — | ns | |
| A/D converter | Trigger input setup time | | $t_{TRGS}$ | 50 | — | 30 | — | ns | Figure 22-35 |

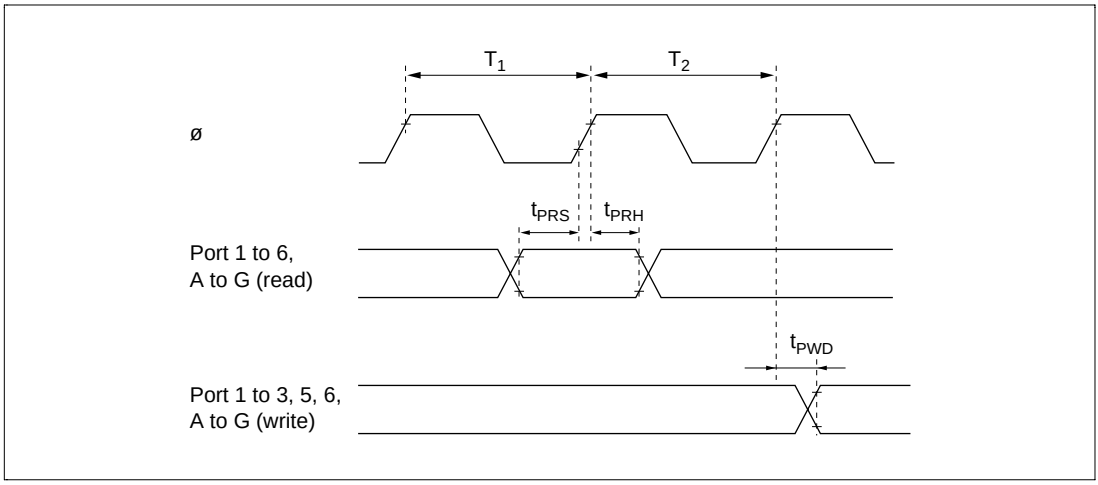



**Figure 22-25 I/O Port Input/Output Timing**

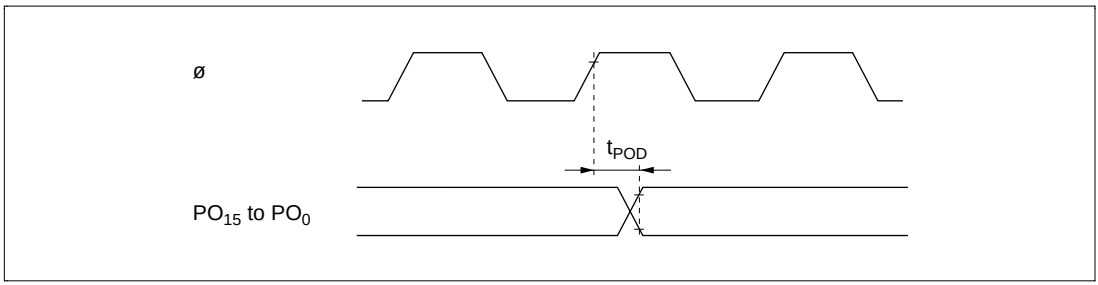
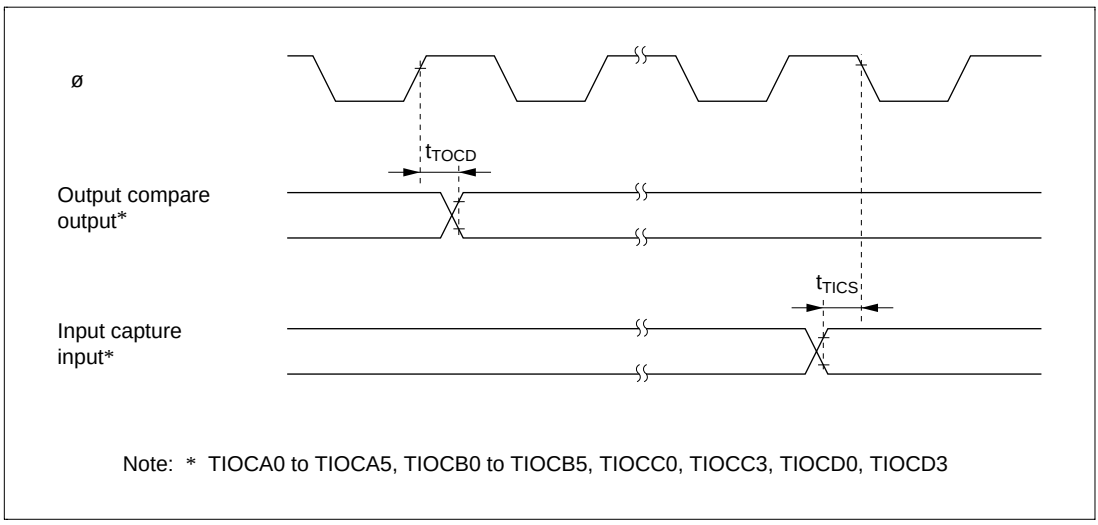



**Figure 22-26 PPG Output Timing**




Note: * TIOCA0 to TIOCA5, TIOCB0 to TIOCB5, TIOCC0, TIOCC3, TIOCD0, TIOCD3

**Figure 22-27 TPU Input/Output Timing**

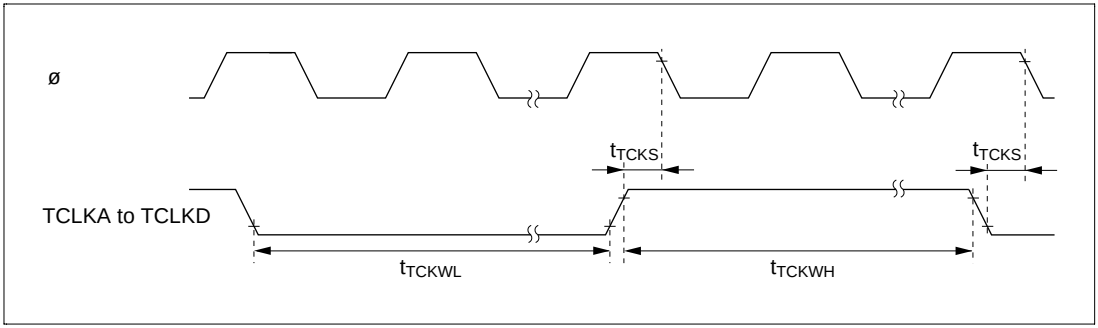



**Figure 22-28 TPU Clock Input Timing**

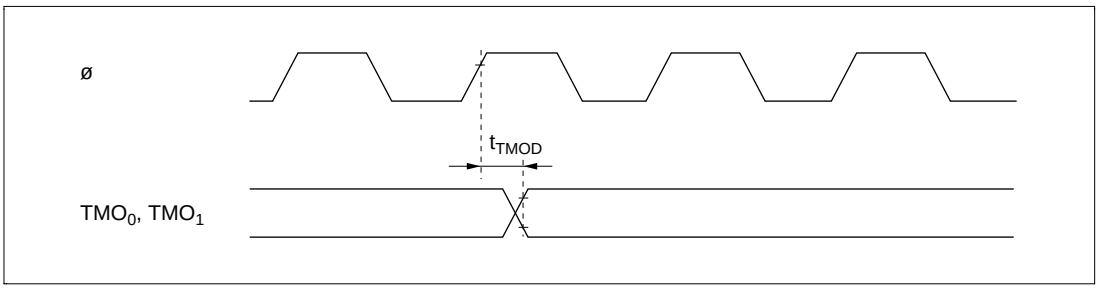



**Figure 22-29 8-Bit Timer Output Timing**

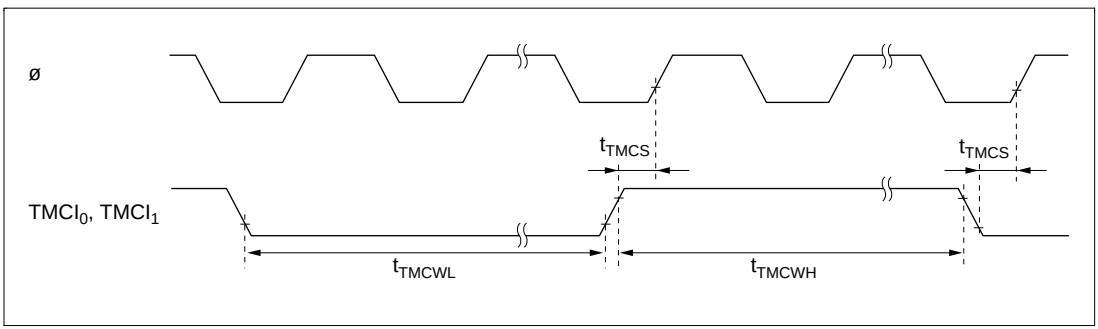



**Figure 22-30 8-Bit Timer Clock Input Timing**

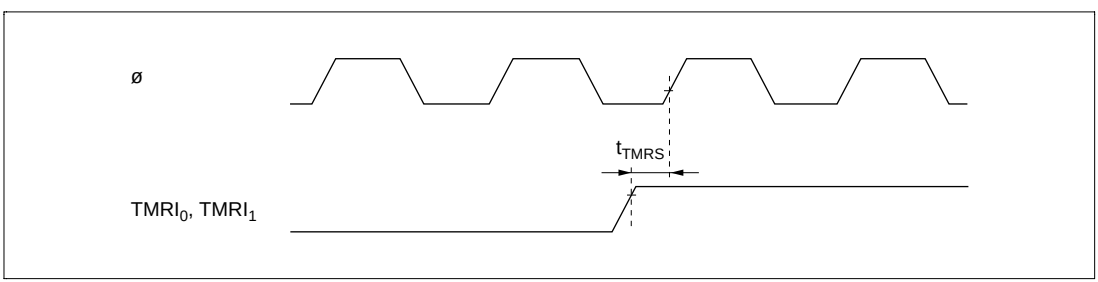



**Figure 22-31 8-Bit Timer Reset Input Timing**

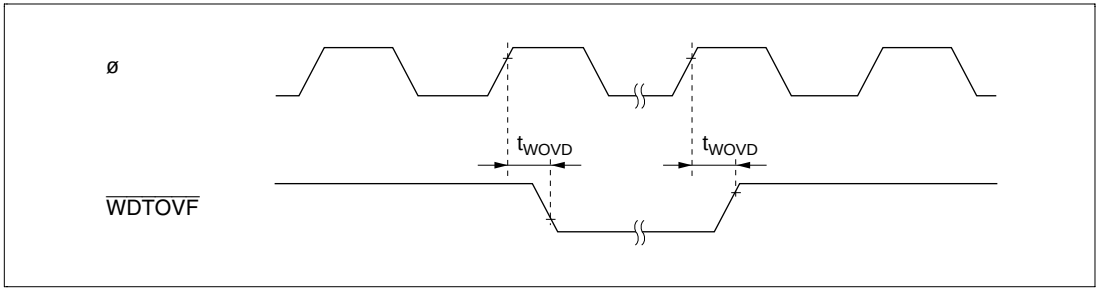



**Figure 22-32 WDT Output Timing**

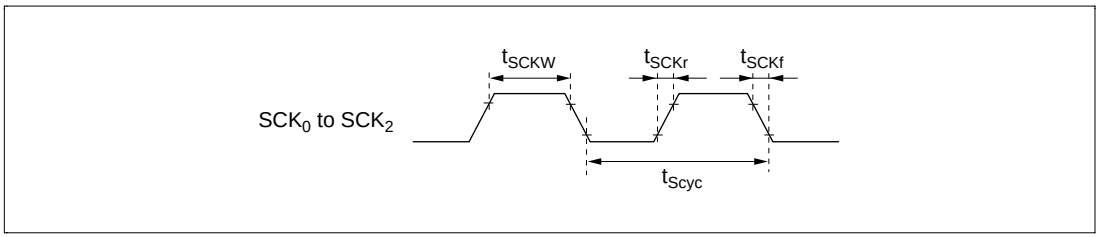



**Figure 22-33 SCK Clock Input Timing**

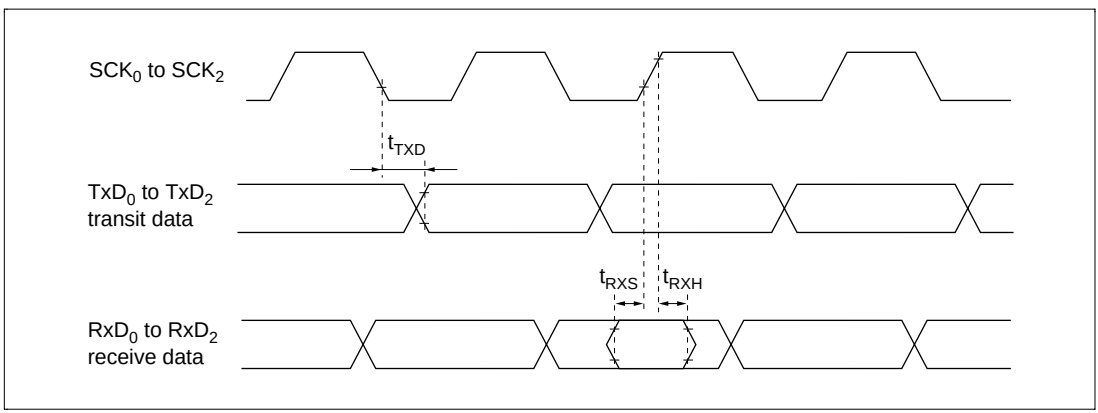



**Figure 22-34 SCI Input/Output Timing (Clock Synchronous Mode)**

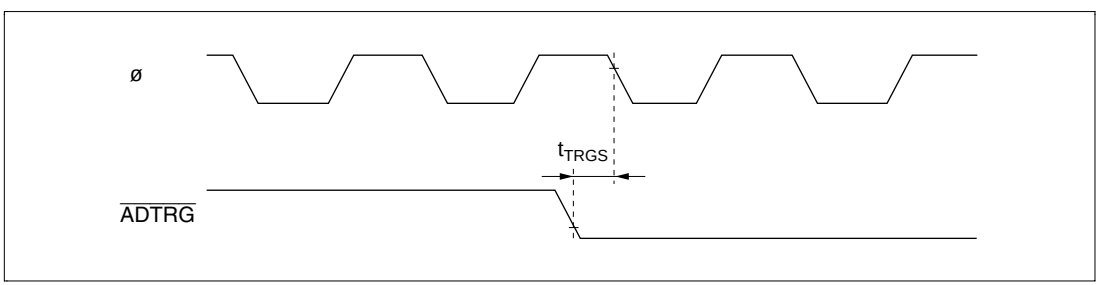



**Figure 22-35 A/D Converter External Trigger Input Timing**

## 22.4 A/D Conversion Characteristics

Table 22-9 lists the A/D conversion characteristics.

**Table 22-9 A/D Conversion Characteristics**

Condition A: $V_{CC} = AV_{CC} = 2.7$ V to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** |
| Resolution | 10 | 10 | 10 | 10 | 10 | 10 | bits |
| Conversion time | — | — | 44 | — | — | 44 | $t_{cyc}$ |
| Analog input capacitance | — | — | 20 | — | — | 20 | pF |
| Permissible signal-source impedance | — | — | 5 | — | — | 5 | kΩ |
| Nonlinearity error | — | — | ±12.0 | — | — | ±8.0 | LSB |
| Offset error | — | — | ±12.0 | — | — | ±8.0 | LSB |
| Full-scale error | — | — | ±12.0 | — | — | ±8.0 | LSB |
| Quantization | — | — | ±0.5 | — | — | ±0.5 | LSB |
| Absolute accuracy | — | — | ±12.0 | — | — | ±8.0 | LSB |

## 22.5 D/A Convervion Characteristics

Table 22-10 lists the D/A conversion characteristics

**Table 22-10 D/A Conversion Characteristics**

Condition A: $V_{CC} = AV_{CC} = 2.7$ V to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | | |
|---|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** | **Test Conditions** |
| Resolution | 8 | 8 | 8 | 8 | 8 | 8 | bit | |
| Conversion time | — | — | 10 | — | — | 10 | µs | 20-pF capacitive load |
| Absolute accuracy | — | ±2.0 | ±3.0 | — | ±1.0 | ±1.5 | LSB | 2-MΩ resistive load |
| | — | — | ±2.0 | — | — | ±1.0 | LSB | 4-MΩ resistive load |

## 22.6 Usage Notes

Although both the ZTAT and mask ROM versions fully meet the electrical specifications listed in this manual, due to differences in the fabrication process, the on-chip ROM, and the layout patterns, there will be differences in the actual values of the electrical characteristics, the operating margins, the noise margins, and other aspects.

Therefore, if a system is evaluated using the ZTAT version, a similar evaluation should also be performed using the mask ROM version.

# Appendix A Instruction Set

## A.1 Instruction List

### Operand Notation

| | |
|---|---|
| Rd | General register (destination)* |
| Rs | General register (source)* |
| Rn | General register* |
| ERn | General register (32-bit register) |
| MAC | Multiply-and-accumulate register (32-bit register) |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extended control register |
| CCR | Condition-code register |
| N | N (negative) flag in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| disp | Displacement |
| + | Add |
| – | Subtract |
| × | Multiply |
| ÷ | Divide |
| ∧ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Logical exclusive OR |
| → | Transfer from the operand on the left to the operand on the right, or transition from the state on the left to the state on the right |
| ¬ | Logical NOT (logical complement) |
| ( ) < > | Contents of operand |
| :8/:16/:24/:32 | 8-, 16-, 24-, or 32-bit length |

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

**Condition Code Notation**

| Symbol | |
|---|---|
| ↕ | Changes according to the result of instruction |
| * | Undetermined (no guaranteed value) |
| 0 | Always cleared to 0 |
| 1 | Always set to 1 |
| — | Not affected by execution of the instruction |

## Table A-1 Instruction Set

### (1) Data Transfer Instructions

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| MOV | MOV.B #xx:8,Rd | B | 2 | | | | | | #xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | MOV.B Rs,Rd | B | 2 | | | | | | Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | MOV.B @ERs,Rd | B | | 2 | | | | | @ERs→Rd8 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.B @(d:16,ERs),Rd | B | | | 4 | | | | @(d:16,ERs)→Rd8 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B @(d:32,ERs),Rd | B | | | 8 | | | | @(d:32,ERs)→Rd8 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.B @ERs+,Rd | B | | | | 2 | | | @ERs→Rd8,ERs32+1→ERs32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B @aa:8,Rd | B | | | | | 2 | | @aa:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.B @aa:16,Rd | B | | | | | 4 | | @aa:16→Rd8 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B @aa:32,Rd | B | | | | | 6 | | @aa:32→Rd8 | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.B Rs,@ERd | B | | 2 | | | | | Rs8→@ERd | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.B Rs,@(d:16,ERd) | B | | | 4 | | | | Rs8→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B Rs,@(d:32,ERd) | B | | | 8 | | | | Rs8→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.B Rs,@-ERd | B | | | | 2 | | | ERd32-1→ERd32,Rs8→@ERd | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B Rs,@aa:8 | B | | | | | 2 | | Rs8→@aa:8 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.B Rs,@aa:16 | B | | | | | 4 | | Rs8→@aa:16 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.B Rs,@aa:32 | B | | | | | 6 | | Rs8→@aa:32 | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.W #xx:16,Rd | W | 4 | | | | | | #xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.W Rs,Rd | W | 2 | | | | | | Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | MOV.W @ERs,Rd | W | | 2 | | | | | @ERs→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 | |

**Table A-1 Instruction Set (cont)**

**(1) Data Transfer Instructions (cont)**

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| MOV | MOV.W @(d:16,ERs),Rd | W | | | 4 | | | | | @(d:16,ERs)→Rd16 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W @(d:32,ERs),Rd | W | | | 8 | | | | | @(d:32,ERs)→Rd16 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.W @ERs+,Rd | W | | | | 2 | | | | @ERs→Rd16,ERs32+2→ERs32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W @aa:16,Rd | W | | | | | 4 | | | @aa:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W @aa:32,Rd | W | | | | | 6 | | | @aa:32→Rd16 | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.W Rs,@ERd | W | | 2 | | | | | | Rs16→@ERd | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.W Rs,@(d:16,ERd) | W | | | 4 | | | | | Rs16→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W Rs,@(d:32,ERd) | W | | | 8 | | | | | Rs16→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.W Rs,@-ERd | W | | | | 2 | | | | ERd32-2→ERd32,Rs16→@ERd | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W Rs,@aa:16 | W | | | | | 4 | | | Rs16→@aa:16 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.W Rs,@aa:32 | W | | | | | 6 | | | Rs16→@aa:32 | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.L #xx:32,ERd | L | 6 | | | | | | | #xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | MOV.L ERs,ERd | L | | 2 | | | | | | ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | MOV.L @ERs,ERd | L | | 4 | | | | | | @ERs→ERd32 | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.L @(d:16,ERs),ERd | L | | | 6 | | | | | @(d:16,ERs)→ERd32 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L @(d:32,ERs),ERd | L | | | 10 | | | | | @(d:32,ERs)→ERd32 | — | — | ↕ | ↕ | 0 | — | 7 | |
| | MOV.L @ERs+,ERd | L | | | | 4 | | | | @ERs→ERd32,ERs32+4→ERs32 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L @aa:16,ERd | L | | | | | 6 | | | @aa:16→ERd32 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L @aa:32,ERd | L | | | | | 8 | | | @aa:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 6 | |

## Table A-1 Instruction Set (cont)

### (1) Data Transfer Instructions (cont)

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| MOV | MOV.L ERs,@ERd | L | | 4 | | | | | | ERs32→@ERd | — | — | ↕ | ↕ | 0 | — | 4 | |
| | MOV.L ERs,@(d:16,ERd) | L | | | 6 | | | | | ERs32→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L ERs,@(d:32,ERd) | L | | | 10 | | | | | ERs32→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 7 | |
| | MOV.L ERs,@-ERd | L | | | | 4 | | | | ERd32-4→ERd32,ERs32→@ERd | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L ERs,@aa:16 | L | | | | | 6 | | | ERs32→@aa:16 | — | — | ↕ | ↕ | 0 | — | 5 | |
| | MOV.L ERs,@aa:32 | L | | | | | 8 | | | ERs32→@aa:32 | — | — | ↕ | ↕ | 0 | — | 6 | |
| POP | POP.W Rn | W | | | | | | | 2 | @SP→Rn16,SP+2→SP | — | — | ↕ | ↕ | 0 | — | 3 | |
| | POP.L ERn | L | | | | | | | 4 | @SP→ERn32,SP+4→SP | — | — | ↕ | ↕ | 0 | — | 5 | |
| PUSH | PUSH.W Rn | W | | | | | | | 2 | SP-2→SP,Rn16→@SP | — | — | ↕ | ↕ | 0 | — | 3 | |
| | PUSH.L ERn | L | | | | | | | 4 | SP-4→SP,ERn32→@SP | — | — | ↕ | ↕ | 0 | — | 5 | |
| LDM | LDM @SP+,(ERm-ERn) | L | | | | | | | 4 | (@SP→ERn32,SP+4→SP)<br>Repeated for each register restored | — | — | — | — | — | — | 7/9/11 [1] | |
| STM | STM (ERm-ERn),@-SP | L | | | | | | | 4 | (SP-4→SP,ERn32→@SP)<br>Repeated for each register saved | — | — | — | — | — | — | 7/9/11 [1] | |
| MOVFPE | MOVFPE @aa:16,Rd | Cannot be used in the H8S/2655 Series | | | | | | | | | | | | | | | [2] | |
| MOVTPE | MOVTPE Rs,@aa:16 | Cannot be used in the H8S/2655 Series | | | | | | | | | | | | | | | [2] | |

## (2) Arithmetic Instructions

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| ADD | ADD.B #xx:8,Rd | B 2 | | | | | | | | Rd8+#xx:8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | ADD.B Rs,Rd | B | 2 | | | | | | | Rd8+Rs8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | ADD.W #xx:16,Rd | W 4 | | | | | | | | Rd16+#xx:16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 | |
| | ADD.W Rs,Rd | W | 2 | | | | | | | Rd16+Rs16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 | |
| | ADD.L #xx:32,ERd | L 6 | | | | | | | | ERd32+#xx:32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 | |
| | ADD.L ERs,ERd | L | 2 | | | | | | | ERd32+ERs32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 | |
| ADDX | ADDX #xx:8,Rd | B 2 | | | | | | | | Rd8+#xx:8+C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 | |
| | ADDX Rs,Rd | B | 2 | | | | | | | Rd8+Rs8+C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 | |
| ADDS | ADDS #1,ERd | L | 2 | | | | | | | ERd32+1→ERd32 | — | — | — | — | — | — | 1 | |
| | ADDS #2,ERd | L | 2 | | | | | | | ERd32+2→ERd32 | — | — | — | — | — | — | 1 | |
| | ADDS #4,ERd | L | 2 | | | | | | | ERd32+4→ERd32 | — | — | — | — | — | — | 1 | |
| INC | INC.B Rd | B | 2 | | | | | | | Rd8+1→Rd8 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | INC.W #1,Rd | W | 2 | | | | | | | Rd16+1→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | INC.W #2,Rd | W | 2 | | | | | | | Rd16+2→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | INC.L #1,ERd | L | 2 | | | | | | | ERd32+1→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | INC.L #2,ERd | L | 2 | | | | | | | ERd32+2→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| DAA | DAA Rd | B | 2 | | | | | | | Rd8 decimal adjust→Rd8 | — | * | ↕ | ↕ | * | ↕ | 1 | |
| SUB | SUB.B Rs,Rd | B | 2 | | | | | | | Rd8-Rs8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | SUB.W #xx:16,Rd | W 4 | | | | | | | | Rd16-#xx:16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 | |

**Table A-1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| SUB | SUB.W Rs,Rd | W | 2 | | | | | | | Rd16-Rs16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 | |
| | SUB.L #xx:32,ERd | L | 6 | | | | | | | ERd32-#xx:32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 | |
| | SUB.L ERs,ERd | L | 2 | | | | | | | ERd32-ERs32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 | |
| SUBX | SUBX #xx:8,Rd | B | 2 | | | | | | | Rd8-#xx:8-C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 | |
| | SUBX Rs,Rd | B | 2 | | | | | | | Rd8-Rs8-C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 | |
| SUBS | SUBS #1,ERd | L | 2 | | | | | | | ERd32-1→ERd32 | — | — | — | — | — | — | 1 | |
| | SUBS #2,ERd | L | 2 | | | | | | | ERd32-2→ERd32 | — | — | — | — | — | — | 1 | |
| | SUBS #4,ERd | L | 2 | | | | | | | ERd32-4→ERd32 | — | — | — | — | — | — | 1 | |
| DEC | DEC.B Rd | B | 2 | | | | | | | Rd8-1→Rd8 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | DEC.W #1,Rd | W | 2 | | | | | | | Rd16-1→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | DEC.W #2,Rd | W | 2 | | | | | | | Rd16-2→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | DEC.L #1,ERd | L | 2 | | | | | | | ERd32-1→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| | DEC.L #2,ERd | L | 2 | | | | | | | ERd32-2→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 | |
| DAS | DAS Rd | B | 2 | | | | | | | Rd8 decimal adjust→Rd8 | — | * | ↕ | ↕ | * | — | 1 | |
| MULXU | MULXU.B Rs,Rd | B | 2 | | | | | | | Rd8×Rs8→Rd16 (unsigned multiplication) | — | — | — | — | — | — | 3 [6] | |
| | MULXU.W Rs,ERd | W | 2 | | | | | | | Rd16×Rs16→ERd32 (unsigned multiplication) | — | — | — | — | — | — | 4 [6] | |
| MULXS | MULXS.B Rs,Rd | B | 4 | | | | | | | Rd8×Rs8→Rd16 (signed multiplication) | — | — | ↕ | ↕ | — | — | 4 [7] | |
| | MULXS.W Rs,ERd | W | 4 | | | | | | | Rd16×Rs16→ERd32 (signed multiplication) | — | — | ↕ | ↕ | — | — | 5 [7] | |

**Table A-1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| DIVXU | DIVXU.B Rs,Rd | B 2 | 2 | | | | | | | Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (unsigned division) | — | — | [8] | [9] | — | — | 12 | |
| | DIVXU.W Rs,ERd | W | 2 | | | | | | | ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (unsigned division) | — | — | [8] | [9] | — | — | 20 | |
| DIVXS | divxs.B Rs,Rd | B 4 | 4 | | | | | | | Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (signed division) | — | — | [10] | [9] | — | — | 13 | |
| | DIVXS.W Rs,ERd | W | 4 | | | | | | | ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (signed division) | — | — | [10] | [9] | — | — | 21 | |
| CMP | CMP.B #xx:8,Rd | B 2 | | | | | | | | Rd8-#xx:8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | CMP.B Rs,Rd | B | 2 | | | | | | | Rd8-Rs8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | CMP.W #xx:16,Rd | W 4 | | | | | | | | Rd16-#xx:16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 | |
| | CMP.W Rs,Rd | W | 2 | | | | | | | Rd16-Rs16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 | |
| | CMP.L #xx:32,ERd | L 6 | | | | | | | | ERd32-#xx:32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 | |
| | CMP.L ERs,ERd | L | 2 | | | | | | | ERd32-ERs32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 | |
| NEG | NEG.B Rd | B | 2 | | | | | | | 0-Rd8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | NEG.W Rd | W | 2 | | | | | | | 0-Rd16→Rd16 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | NEG.L ERd | L | 2 | | | | | | | 0-ERd32→ERd32 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| EXTU | EXTU.W Rd | W | 2 | | | | | | | 0→(<bit 15 to 8> of Rd16) | — | — | 0 | ↕ | 0 | — | 1 | |
| | EXTU.L ERd | L | 2 | | | | | | | 0→(<bit 31 to 16> of ERd32) | — | — | 0 | ↕ | 0 | — | 1 | |

**Table A-1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | | I | H | N | Z | V | C | Normal | Advanced |
| EXTS | EXTS.W Rd | W | 2 | | | | | | | | | (<bit 7> of Rd16)→ (<bit 15 to 8> of Rd16) | — | — | ↕ | ↕ | 0 | — | 1 | |
| | EXTS.L ERd | L | 2 | | | | | | | | | (<bit 15> of ERd32)→ (<bit 31 to 16> of ERd32) | — | — | ↕ | ↕ | 0 | — | 1 | |
| TAS | TAS @ERd | B | | 4 | | | | | | | | @ERd-0→CCR set, (1)→ (<bit 7> of @ERd) | — | — | ↕ | ↕ | 0 | — | 4 | |
| MAC | MAC @ERn+,@ERm+ | — | | | | 4 | | | | | | @ERn×@ERm+MAC→MAC (signed multiplication) ERn+2→ERn,ERm+2→ERm | — | — | — | — | — | — | 4 | |
| | | | | | | | | | | | | | | | [11] | [11] | [11] | | | |
| CLRMAC | CLRMAC | — | | | | | | | | 2 | | 0→MACH,MACL | — | — | — | — | — | — | 2 [12] | |
| LDMAC | LDMAC ERs,MACH | L | | 2 | | | | | | | | ERs→MACH | — | — | — | — | — | — | 2 [12] | |
| | LDMAC ERs,MACL | L | | 2 | | | | | | | | ERs→MACL | — | — | — | — | — | — | 2 [12] | |
| STMAC | STMAC MACH,ERd | L | | 2 | | | | | | | | MACH→ERd | — | — | ↕ | ↕ | ↕ | — | 1 [12] | |
| | STMAC MACL,ERd | L | | 2 | | | | | | | | MACL→ERd | — | — | ↕ | ↕ | ↕ | — | 1 [12] | |

## (3) Logical Instructions

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| AND | AND.B #xx:8,Rd | B 2 | | | | | | | | Rd8∧#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | AND.B Rs,Rd | B | 2 | | | | | | | Rd8∧Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | AND.W #xx:16,Rd | W 4 | | | | | | | | Rd16∧#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | AND.W Rs,Rd | W | 2 | | | | | | | Rd16∧Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | AND.L #xx:32,ERd | L 6 | | | | | | | | ERd32∧#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | AND.L ERs,ERd | L | 4 | | | | | | | ERd32∧ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 | |
| OR | OR.B #xx:8,Rd | B 2 | | | | | | | | Rd8∨#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | OR.B Rs,Rd | B | 2 | | | | | | | Rd8∨Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | OR.W #xx:16,Rd | W 4 | | | | | | | | Rd16∨#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | OR.W Rs,Rd | W | 2 | | | | | | | Rd16∨Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | OR.L #xx:32,ERd | L 6 | | | | | | | | ERd32∨#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | OR.L ERs,ERd | L | 4 | | | | | | | ERd32∨ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 | |
| XOR | XOR.B #xx:8,Rd | B 2 | | | | | | | | Rd8⊕#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | XOR.B Rs,Rd | B | 2 | | | | | | | Rd8⊕Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | XOR.W #xx:16,Rd | W 4 | | | | | | | | Rd16⊕#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | XOR.W Rs,Rd | W | 2 | | | | | | | Rd16⊕Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | XOR.L #xx:32,ERd | L 6 | | | | | | | | ERd32⊕#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 | |
| | XOR.L ERs,ERd | L | 4 | | | | | | | ERd32⊕ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 | |
| NOT | NOT.B Rd | B | 2 | | | | | | | ¬ Rd8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | NOT.W Rd | W | 2 | | | | | | | ¬ Rd16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 | |
| | NOT.L ERd | L | 2 | | | | | | | ¬ ERd32→ERd32 | — | — | ↕ | ↕ | 0 | — | 1 | |

**Table A-1 Instruction Set**

**(4) Shift Instructions**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | | I | H | N | Z | V | C | Normal | Advanced |
| SHAL | SHAL.B Rd | B | | 2 | | | | | | | |  | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| | SHAL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| | SHAL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| | SHAL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| | SHAL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| | SHAL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | | 1 |
| SHAR | SHAR.B Rd | B | | 2 | | | | | | | |  | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHAR.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHAR.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHAR.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHAR.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHAR.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| SHLL | SHLL.B Rd | B | | 2 | | | | | | | |  | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHLL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHLL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHLL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHLL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |
| | SHLL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | | 1 |

## (4) Shift Instructions (cont)

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | | | I | H | N | Z | V | C | Normal | Advanced |
| SHLR | SHLR.B Rd | B | | 2 | | | | | | | | | 0 → MSB → LSB → C | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| | SHLR.B #2,Rd | B | | 2 | | | | | | | | | | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| | SHLR.W Rd | W | | 2 | | | | | | | | | | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| | SHLR.W #2,Rd | W | | 2 | | | | | | | | | | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| | SHLR.L ERd | L | | 2 | | | | | | | | | | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| | SHLR.L #2,ERd | L | | 2 | | | | | | | | | | — | — | 0 ↔ | 0 ↔ | 0 | ↔ | 1 | |
| ROTXL | ROTXL.B Rd | B | | 2 | | | | | | | | | C → MSB → LSB → C | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXL.B #2,Rd | B | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXL.W Rd | W | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXL.W #2,Rd | W | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXL.L ERd | L | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXL.L #2,ERd | L | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| ROTXR | ROTXR.B Rd | B | | 2 | | | | | | | | | MSB → LSB → C | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXR.B #2,Rd | B | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXR.W Rd | W | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXR.W #2,Rd | W | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXR.L ERd | L | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |
| | ROTXR.L #2,ERd | L | | 2 | | | | | | | | | | — | — | ↔ | ↔ | 0 | ↔ | 1 | |

## (4) Shift Instructions (cont)

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | | | I | H | N | Z | V | C | Normal | Advanced |
| ROTL | ROTL.B Rd | B | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTL.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTL.W Rd | W | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTL.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTL.L ERd | L | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTL.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| ROTR | ROTR.B Rd | B | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTR.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTR.W Rd | W | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTR.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTR.L ERd | L | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |
| | ROTR.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 | |

## (5) Bit-Manipulation Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| BSET | BSET #xx:3,Rd | B | | 2 | | | | | | | (#xx:3 of Rd8)←1 | — | — | — | — | — | — | 1 | |
| | BSET #xx:3,@ERd | B | | | 4 | | | | | | (#xx:3 of @ERd)←1 | — | — | — | — | — | — | 4 | |
| | BSET #xx:3,@aa:8 | B | | | | | | 4 | | | (#xx:3 of @aa:8)←1 | — | — | — | — | — | — | 4 | |
| | BSET #xx:3,@aa:16 | B | | | | | | 6 | | | (#xx:3 of @aa:16)←1 | — | — | — | — | — | — | 5 | |
| | BSET #xx:3,@aa:32 | B | | | | | | 8 | | | (#xx:3 of @aa:32)←1 | — | — | — | — | — | — | 6 | |
| | BSET Rn,Rd | B | | 2 | | | | | | | (Rn8 of Rd8)←1 | — | — | — | — | — | — | 1 | |
| | BSET Rn,@ERd | B | | | 4 | | | | | | (Rn8 of @ERd)←1 | — | — | — | — | — | — | 4 | |
| | BSET Rn,@aa:8 | B | | | | | | 4 | | | (Rn8 of @aa:8)←1 | — | — | — | — | — | — | 4 | |
| | BSET Rn,@aa:16 | B | | | | | | 6 | | | (Rn8 of @aa:16)←1 | — | — | — | — | — | — | 5 | |
| | BSET Rn,@aa:32 | B | | | | | | 8 | | | (Rn8 of @aa:32)←1 | — | — | — | — | — | — | 6 | |
| BCLR | BCLR #xx:3,Rd | B | | 2 | | | | | | | (#xx:3 of Rd8)←0 | — | — | — | — | — | — | 1 | |
| | BCLR #xx:3,@ERd | B | | | 4 | | | | | | (#xx:3 of @ERd)←0 | — | — | — | — | — | — | 4 | |
| | BCLR #xx:3,@aa:8 | B | | | | | | 4 | | | (#xx:3 of @aa:8)←0 | — | — | — | — | — | — | 4 | |
| | BCLR #xx:3,@aa:16 | B | | | | | | 6 | | | (#xx:3 of @aa:16)←0 | — | — | — | — | — | — | 5 | |
| | BCLR #xx:3,@aa:32 | B | | | | | | 8 | | | (#xx:3 of @aa:32)←0 | — | — | — | — | — | — | 6 | |
| | BCLR Rn,Rd | B | | 2 | | | | | | | (Rn8 of Rd8)←0 | — | — | — | — | — | — | 1 | |
| | BCLR Rn,@ERd | B | | | 4 | | | | | | (Rn8 of @ERd)←0 | — | — | — | — | — | — | 4 | |
| | BCLR Rn,@aa:8 | B | | | | | | 4 | | | (Rn8 of @aa:8)←0 | — | — | — | — | — | — | 4 | |
| | BCLR Rn,@aa:16 | B | | | | | | 6 | | | (Rn8 of @aa:16)←0 | — | — | — | — | — | — | 5 | |

## (5) Bit-Manipulation Instructions (cont)

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| BCLR | BCLR Rn,@aa:32 | B | | | | | 8 | | | (Rn8 of @aa:32)←0 | — | — | — | ↕ | — | — | 6 | |
| BNOT | BNOT #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)←[¬ (#xx:3 of Rd8)] | — | — | — | ↕ | — | — | 1 | |
| | BNOT #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd)←<br>[¬ (#xx:3 of @ERd)] | — | — | — | ↕ | — | — | 4 | |
| | BNOT #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)←<br>[¬ (#xx:3 of @aa:8)] | — | — | — | ↕ | — | — | 4 | |
| | BNOT #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)←<br>[¬ (#xx:3 of @aa:16)] | — | — | — | ↕ | — | — | 5 | |
| | BNOT #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)←<br>[¬ (#xx:3 of @aa:32)] | — | — | — | ↕ | — | — | 6 | |
| | BNOT Rn,Rd | B | 2 | | | | | | | (Rn8 of Rd8)←[¬ (Rn8 of Rd8)] | — | — | — | ↕ | — | — | 1 | |
| | BNOT Rn,@ERd | B | | 4 | | | | | | (Rn8 of @ERd)←[¬ (Rn8 of @ERd)] | — | — | — | ↕ | — | — | 4 | |
| | BNOT Rn,@aa:8 | B | | | | | 4 | | | (Rn8 of @aa:8)←[¬ (Rn8 of @aa:8)] | — | — | — | ↕ | — | — | 4 | |
| | BNOT Rn,@aa:16 | B | | | | | 6 | | | (Rn8 of @aa:16)←<br>[¬ (Rn8 of @aa:16)] | — | — | — | ↕ | — | — | 5 | |
| | BNOT Rn,@aa:32 | B | | | | | 8 | | | (Rn8 of @aa:32)←<br>[¬ (Rn8 of @aa:32)] | — | — | — | ↕ | — | — | 6 | |
| BTST | BTST #xx:3,Rd | B | 2 | | | | | | | ¬ (#xx:3 of Rd8)→Z | — | — | — | ↕ | — | — | 1 | |
| | BTST #xx:3,@ERd | B | | 4 | | | | | | ¬ (#xx:3 of @ERd)→Z | — | — | — | ↕ | — | — | 3 | |
| | BTST #xx:3,@aa:8 | B | | | | | 4 | | | ¬ (#xx:3 of @aa:8)→Z | — | — | — | ↕ | — | — | 3 | |
| | BTST #xx:3,@aa:16 | B | | | | | 6 | | | ¬ (#xx:3 of @aa:16)→Z | — | — | — | ↕ | — | — | 4 | |

## (5) Bit-Manipulation Instructions (cont)

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| BTST | BTST #xx:3,@aa:32 | B | | | | | 8 | | | ¬ (#xx:3 of @aa:32)→Z | — | — | ↕ | ↕ | — | — | 5 | |
| | BTST Rn,Rd | B | 2 | | | | | | | ¬ (Rn8 of Rd8)→Z | — | — | ↕ | ↕ | — | — | 1 | |
| | BTST Rn,@ERd | B | | 4 | | | | | | ¬ (Rn8 of @ERd)→Z | — | — | ↕ | ↕ | — | — | 3 | |
| | BTST Rn,@aa:8 | B | | | | | 4 | | | ¬ (Rn8 of @aa:8)→Z | — | — | ↕ | ↕ | — | — | 3 | |
| | BTST Rn,@aa:16 | B | | | | | 6 | | | ¬ (Rn8 of @aa:16)→Z | — | — | ↕ | ↕ | — | — | 4 | |
| | BTST Rn,@aa:32 | B | | | | | 8 | | | ¬ (Rn8 of @aa:32)→Z | — | — | ↕ | ↕ | — | — | 5 | |
| BLD | BLD #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 | |
| | BLD #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd) →C | — | — | — | — | — | ↕ | 3 | |
| | BLD #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 | |
| | BLD #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 | |
| | BLD #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 | |
| BILD | BILD #xx:3,Rd | B | 2 | | | | | | | ¬ (#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 | |
| | BILD #xx:3,@ERd | B | | 4 | | | | | | ¬ (#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 | |
| | BILD #xx:3,@aa:8 | B | | | | | 4 | | | ¬ (#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 | |
| | BILD #xx:3,@aa:16 | B | | | | | 6 | | | ¬ (#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 | |
| | BILD #xx:3,@aa:32 | B | | | | | 8 | | | ¬ (#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 | |
| BST | BST #xx:3,Rd | B | 2 | | | | | | | C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 | |
| | BST #xx:3,@ERd | B | | 4 | | | | | | C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 | |
| | BST #xx:3,@aa:8 | B | | | | | 4 | | | C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 | |

**Table A-1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | Operation | Condition Code | | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| BST | BST #xx:3,@aa:16 | B | | | | | 6 | | | C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 | |
| | BST #xx:3,@aa:32 | B | | | | | 8 | | | C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 | |
| BIST | BIST #xx:3,Rd | B | 2 | | | | | | | ¬ C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 | |
| | BIST #xx:3,@ERd | B | | 4 | | | | | | ¬ C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 | |
| | BIST #xx:3,@aa:8 | B | | | | | 4 | | | ¬ C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 | |
| | BIST #xx:3,@aa:16 | B | | | | | 6 | | | ¬ C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 | |
| | BIST #xx:3,@aa:32 | B | | | | | 8 | | | ¬ C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 | |
| BAND | BAND #xx:3,Rd | B | 2 | | | | | | | C∧(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 | |
| | BAND #xx:3,@ERd | B | | 4 | | | | | | C∧(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 | |
| | BAND #xx:3,@aa:8 | B | | | | | 4 | | | C∧(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 | |
| | BAND #xx:3,@aa:16 | B | | | | | 6 | | | C∧(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 | |
| | BAND #xx:3,@aa:32 | B | | | | | 8 | | | C∧(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 | |
| BIAND | BIAND #xx:3,Rd | B | 2 | | | | | | | C∧[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 | |
| | BIAND #xx:3,@ERd | B | | 4 | | | | | | C∧[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIAND #xx:3,@aa:8 | B | | | | | 4 | | | C∧[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIAND #xx:3,@aa:16 | B | | | | | 6 | | | C∧[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 | |
| | BIAND #xx:3,@aa:32 | B | | | | | 8 | | | C∧[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 | |
| BOR | BOR #xx:3,Rd | B | 2 | | | | | | | C∨(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 | |
| | BOR #xx:3,@ERd | B | | 4 | | | | | | C∨(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 | |

## (5) Bit-Manipulation Instructions (cont)

| Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| BOR | BOR #xx:3,@aa:8 | B | | | | | 4 | | | C∨(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 | |
| | BOR #xx:3,@aa:16 | B | | | | | 6 | | | C∨(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 | |
| | BOR #xx:3,@aa:32 | B | | | | | 8 | | | C∨(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 | |
| BIOR | BIOR #xx:3,Rd | B | 2 | | | | | | | C∨[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 | |
| | BIOR #xx:3,@ERd | B | | 4 | | | | | | C∨[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIOR #xx:3,@aa:8 | B | | | | | 4 | | | C∨[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIOR #xx:3,@aa:16 | B | | | | | 6 | | | C∨[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 | |
| | BIOR #xx:3,@aa:32 | B | | | | | 8 | | | C∨[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 | |
| BXOR | BXOR #xx:3,Rd | B | 2 | | | | | | | C⊕(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 | |
| | BXOR #xx:3,@ERd | B | | 4 | | | | | | C⊕(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 | |
| | BXOR #xx:3,@aa:8 | B | | | | | 4 | | | C⊕(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 | |
| | BXOR #xx:3,@aa:16 | B | | | | | 6 | | | C⊕(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 | |
| | BXOR #xx:3,@aa:32 | B | | | | | 8 | | | C⊕(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 | |
| BIXOR | BIXOR #xx:3,Rd | B | 2 | | | | | | | C⊕[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 | |
| | BIXOR #xx:3,@ERd | B | | 4 | | | | | | C⊕[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIXOR #xx:3,@aa:8 | B | | | | | 4 | | | C⊕[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 | |
| | BIXOR #xx:3,@aa:16 | B | | | | | 6 | | | C⊕[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 | |
| | BIXOR #xx:3,@aa:32 | B | | | | | 8 | | | C⊕[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 | |

## Table A-1 Instruction Set

### (6) Branch Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Branching Condition | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | | I | H | N | Z | V | C | Normal | Advanced |
| Bcc | BRA d:8(BT d:8) | — | | | | | | | 2 | | | if condition is true then PC←PC+d else next; | Always | — | — | — | — | — | — | 2 | |
| | BRA d:16(BT d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BRN d:8(BF d:8) | — | | | | | | | 2 | | | | Never | — | — | — | — | — | — | 2 | |
| | BRN d:16(BF d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BHI d:8 | — | | | | | | | 2 | | | | C∨Z=0 | — | — | — | — | — | — | 2 | |
| | BHI d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BLS d:8 | — | | | | | | | 2 | | | | C∨Z=1 | — | — | — | — | — | — | 2 | |
| | BLS d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BCC d:8(BHS d:8) | — | | | | | | | 2 | | | | C=0 | — | — | — | — | — | — | 2 | |
| | BCC d:16(BHS d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BCS d:8(BLO d:8) | — | | | | | | | 2 | | | | C=1 | — | — | — | — | — | — | 2 | |
| | BCS d:16(BLO d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BNE d:8 | — | | | | | | | 2 | | | | Z=0 | — | — | — | — | — | — | 2 | |
| | BNE d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BEQ d:8 | — | | | | | | | 2 | | | | Z=1 | — | — | — | — | — | — | 2 | |
| | BEQ d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |
| | BVC d:8 | — | | | | | | | 2 | | | | V=0 | — | — | — | — | — | — | 2 | |
| | BVC d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 | |

**Table A-1 Instruction Set (cont)**

**(6) Branch Instructions (cont)**

| | Mnemonic | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | Branching Condition | I | H | N | Z | V | C | Normal | Advanced |
| Bcc | BVS d:8 | — | | | | | | | 2 | | | V=1 | — | — | — | — | — | — | 2 | |
| | BVS d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BPL d:8 | — | | | | | | | 2 | | | N=0 | — | — | — | — | — | — | 2 | |
| | BPL d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BMI d:8 | — | | | | | | | 2 | | | N=1 | — | — | — | — | — | — | 2 | |
| | BMI d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BGE d:8 | — | | | | | | | 2 | | | N⊕V=0 | — | — | — | — | — | — | 2 | |
| | BGE d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BLT d:8 | — | | | | | | | 2 | | | N⊕V=1 | — | — | — | — | — | — | 2 | |
| | BLT d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BGT d:8 | — | | | | | | | 2 | | | Z∨(N⊕V)=0 | — | — | — | — | — | — | 2 | |
| | BGT d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |
| | BLE d:8 | — | | | | | | | 2 | | | Z∨(N⊕V)=1 | — | — | — | — | — | — | 2 | |
| | BLE d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 | |

**Table A-1 Instruction Set (cont)**

**(6) Branch Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | I | H | N | Z | V | C | Normal | Advanced |
| JMP | JMP @ERn | — | | | 2 | | | | | | PC←ERn | — | — | — | — | — | — | 2 | |
| | JMP @aa:24 | — | | | | | | 4 | | | PC←aa:24 | — | — | — | — | — | — | 3 | |
| | JMP @@aa:8 | — | | | | | | | | 2 | PC←@aa:8 | — | — | — | — | — | — | 4 | 5 |
| BSR | BSR d:8 | — | | | | | | | 2 | | PC→@-SP,PC←PC+d:8 | — | — | — | — | — | — | 3 | 4 |
| | BSR d:16 | — | | | | | | | 4 | | PC→@-SP,PC←PC+d:16 | — | — | — | — | — | — | 4 | 5 |
| JSR | JSR @ERn | — | | | 2 | | | | | | PC→@-SP,PC←ERn | — | — | — | — | — | — | 3 | 4 |
| | JSR @aa:24 | — | | | | | | 4 | | | PC→@-SP,PC←aa:24 | — | — | — | — | — | — | 4 | 5 |
| | JSR @@aa:8 | — | | | | | | | | 2 | PC→@-SP,PC←@aa:8 | — | — | — | — | — | — | 4 | 6 |
| RTS | RTS | — | | | | | | | | 2 | PC←@SP+ | — | — | — | — | — | — | 4 | 5 |

## (7) System Control Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @@aa | | I | H | N | Z | V | C | Normal | Advanced |
| TRAPA | TRAPA #xx:2 | — | | | | | | | | PC→@-SP,CCR→@-SP,<br>EXR→@-SP,<vector>→PC | 1 | — | — | — | — | — | 7 [13] | 8 [13] |
| RTE | RTE | — | | | | | | | | EXR←@SP+,CCR←@SP+,<br>PC←@SP+ | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 5 [13] | |
| SLEEP | SLEEP | — | | | | | | | | Transition to power-down state | — | — | — | — | — | — | 2 | |
| LDC | LDC #xx:8,CCR | B | 2 | | | | | | | #xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | LDC #xx:8,EXR | B | 4 | | | | | | | #xx:8→EXR | — | — | — | — | — | — | 2 | |
| | LDC Rs,CCR | B | | 2 | | | | | | Rs8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | LDC Rs,EXR | B | | 2 | | | | | | Rs8→EXR | — | — | — | — | — | — | 1 | |
| | LDC @ERs,CCR | W | | | 4 | | | | | @ERs→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 3 | |
| | LDC @ERs,EXR | W | | | 4 | | | | | @ERs→EXR | — | — | — | — | — | — | 3 | |
| | LDC @(d:16,ERs),CCR | W | | | | 6 | | | | @(d:16,ERs)→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 | |
| | LDC @(d:16,ERs),EXR | W | | | | 6 | | | | @(d:16,ERs)→EXR | — | — | — | — | — | — | 4 | |
| | LDC @(d:32,ERs),CCR | W | | | | 10 | | | | @(d:32,ERs)→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 6 | |
| | LDC @(d:32,ERs),EXR | W | | | | 10 | | | | @(d:32,ERs)→EXR | — | — | — | — | — | — | 6 | |
| | LDC @ERs+,CCR | W | | | | | 4 | | | @ERs→CCR,ERs32+2→ERs32 | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 | |
| | LDC @ERs+,EXR | W | | | | | 4 | | | @ERs→EXR,ERs32+2→ERs32 | — | — | — | — | — | — | 4 | |
| | LDC @aa:16,CCR | W | | | | | | 6 | | @aa:16→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 | |
| | LDC @aa:16,EXR | W | | | | | | 6 | | @aa:16→EXR | — | — | — | — | — | — | 4 | |
| | LDC @aa:32,CCR | W | | | | | | 8 | | @aa:32→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 5 | |
| | LDC @aa:32,EXR | W | | | | | | 8 | | @aa:32→EXR | — | — | — | — | — | — | 5 | |

**Table A-1 Instruction Set (cont)**

**(7) System Control Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Normal | Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| STC | STC CCR,Rd | B | | 2 | | | | | | | | CCR→Rd8 | — | — | — | — | — | — | 1 | |
| | STC EXR,Rd | B | | 2 | | | | | | | | EXR→Rd8 | — | — | — | — | — | — | 1 | |
| | STC CCR,@ERd | W | | | 4 | | | | | | | CCR→@ERd | — | — | — | — | — | — | 3 | |
| | STC EXR,@ERd | W | | | 4 | | | | | | | EXR→@ERd | — | — | — | — | — | — | 3 | |
| | STC CCR,@(d:16,ERd) | W | | | | 6 | | | | | | CCR→@(d:16,ERd) | — | — | — | — | — | — | 4 | |
| | STC EXR,@(d:16,ERd) | W | | | | 6 | | | | | | EXR→@(d:16,ERd) | — | — | — | — | — | — | 4 | |
| | STC CCR,@(d:32,ERd) | W | | | | 10 | | | | | | CCR→@(d:32,ERd) | — | — | — | — | — | — | 6 | |
| | STC EXR,@(d:32,ERd) | W | | | | 10 | | | | | | EXR→@(d:32,ERd) | — | — | — | — | — | — | 6 | |
| | STC CCR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,CCR→@ERd | — | — | — | — | — | — | 4 | |
| | STC EXR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,EXR→@ERd | — | — | — | — | — | — | 4 | |
| | STC CCR,@aa:16 | W | | | | | | 6 | | | | CCR→@aa:16 | — | — | — | — | — | — | 4 | |
| | STC EXR,@aa:16 | W | | | | | | 6 | | | | EXR→@aa:16 | — | — | — | — | — | — | 4 | |
| | STC CCR,@aa:32 | W | | | | | | 8 | | | | CCR→@aa:32 | — | — | — | — | — | — | 5 | |
| | STC EXR,@aa:32 | W | | | | | | 8 | | | | EXR→@aa:32 | — | — | — | — | — | — | 5 | |
| ANDC | ANDC #xx:8,CCR | B | 2 | | | | | | | | | CCR∧#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | ANDC #xx:8,EXR | B | 4 | | | | | | | | | EXR∧#xx:8→EXR | — | — | — | — | — | — | 2 | |
| ORC | ORC #xx:8,CCR | B | 2 | | | | | | | | | CCR∨#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | ORC #xx:8,EXR | B | 4 | | | | | | | | | EXR∨#xx:8→EXR | — | — | — | — | — | — | 2 | |
| XORC | XORC #xx:8,CCR | B | 2 | | | | | | | | | CCR⊕#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 | |
| | XORC #xx:8,EXR | B | 4 | | | | | | | | | EXR⊕#xx:8→EXR | — | — | — | — | — | — | 2 | |
| NOP | NOP | — | | | | | | | | | 2 | PC←PC+2 | — | — | — | — | — | — | 1 | |

## (8) Block Transfer Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | | Operation | Condition Code | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | | | I | H | N | Z | V | C | Normal | Advanced |
| EEPMOV | EEPMOV.B | — | | | | | | | | | 4 | if R4L≠0<br>Repeat @ER5→@ER6<br>ER5+1→ER5<br>ER6+1→ER6<br>R4L-1→R4L<br>Until R4L=0<br>else next; | — | — | — | — | — | — | 4+2n *2 | |
| | EEPMOV.W | — | | | | | | | | | 4 | if R4≠0<br>Repeat @ER5→@ER6<br>ER5+1→ER5<br>ER6+1→ER6<br>R4-1→R4<br>Until R4=0<br>else next; | — | — | — | — | — | — | 4+2n *2 | |

Notes: 1. The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.
2. n is the initial value of R4L or R4.
[1] Seven states for saving or restoring two registers, nine states for three registers, or eleven states for four registers.
[2] Cannot be used in the H8S/2655 Series.
[3] Set to 1 when a carry or borrow occurs at bit 11; otherwise cleared to 0.
[4] Set to 1 when a carry or borrow occurs at bit 27; otherwise cleared to 0.
[5] Retains its previous value when the result is zero; otherwise cleared to 0.

[6] One additional state is required for execution immediately after a MULXU, MULXS, or STMAC instruction. Also, a maximum of three additional states are required for execution of a MULXU instruction within three states after execution of a MAC instruction. For example, if there is a one-state instruction (such as NOP) between a MAC instruction and a MULXU instruction, the MULXU instruction will be two states longer.

[7] A maximum of two additional states are required for execution of a MULXS instruction within two states after execution of a MAC instruction. For example, if there is a one-state instruction (such as NOP) between a MAC instruction and a MULXS instruction, the MULXS instruction will be one state longer.

[8] Set to 1 when the divisor is negative; otherwise cleared to 0.

[9] Set to 1 when the divisor is zero; otherwise cleared to 0.

[10] Set to 1 when the quotient is negative; otherwise cleared to 0.

[11] MAC instruction results are indicated in the flags when the STMAC instruction is executed.

[12] A maximum of three additional states are required for execution of one of these instructions within three states after execution of a MAC instruction. For example, if there is a one-state instruction (such as NOP) between a MAC instruction and one of these instructions, that instruction will be two states longer.

[13] One additional state is required for execution when EXR is valid.

## A.2 Instruction Codes

Table A-2 shows the instruction codes.

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| ADD | ADD.B #xx:8,Rd | B | 8 rd | IMM | | | | | | | | |
| | ADD.B Rs,Rd | B | 0 8 | rs rd | | | | | | | | |
| | ADD.W #xx:16,Rd | W | 7 9 | 1 rd | IMM | | | | | | | |
| | ADD.W Rs,Rd | W | 0 9 | rs rd | | | | | | | | |
| | ADD.L #xx:32,ERd | L | 7 A | 1 0 erd | IMM | | | | | | | |
| | ADD.L ERs,ERd | L | 0 A | 1 ers 0 erd | | | | | | | | |
| ADDS | ADDS #1,ERd | L | 0 B | 0 0 erd | | | | | | | | |
| | ADDS #2,ERd | L | 0 B | 8 0 erd | | | | | | | | |
| | ADDS #4,ERd | L | 0 B | 9 0 erd | | | | | | | | |
| ADDX | ADDX #xx:8,Rd | B | 9 rd | IMM | | | | | | | | |
| | ADDX Rs,Rd | B | 0 E | rs rd | | | | | | | | |
| AND | AND.B #xx:8,Rd | B | E rd | IMM | | | | | | | | |
| | AND.B Rs,Rd | B | 1 6 | rs rd | | | | | | | | |
| | AND.W #xx:16,Rd | W | 7 9 | 6 rd | IMM | | | | | | | |
| | AND.W Rs,Rd | W | 6 6 | rs rd | | | | | | | | |
| | AND.L #xx:32,ERd | L | 7 A | 6 0 erd | IMM | | | | | | | |
| | AND.L ERs,ERd | L | 0 1 | F 0 | 6 6 | 0 ers 0 erd | | | | | | |
| ANDC | ANDC #xx:8,CCR | B | 0 6 | IMM | | | | | | | | |
| | ANDC #xx:8,EXR | B | 0 1 | 4 1 | 0 6 | IMM | | | | | | |
| BAND | BAND #xx:3,Rd | B | 7 6 | 0 IMM rd | | | | | | | | |
| | BAND #xx:3,@ERd | B | 7 C | 0 erd 0 | 7 6 | 0 IMM 0 | | | | | | |
| | BAND #xx:3,@aa:8 | B | 7 E | abs | 7 6 | 0 IMM 0 | | | | | | |
| | BAND #xx:3,@aa:16 | B | 6 A | 1 0 | abs | | 7 6 | 0 IMM 0 | | | | |
| | BAND #xx:3,@aa:32 | B | 6 A | 3 0 | abs | | | | 7 6 | 0 IMM 0 | | |
| Bcc | BRA d:8 (BT d:8) | — | 4 0 | disp | | | | | | | | |
| | BRA d:16 (BT d:16) | — | 5 8 | 0 0 | disp | | | | | | | |
| | BRN d:8 (BF d:8) | — | 4 1 | disp | | | | | | | | |
| | BRN d:16 (BF d:16) | — | 5 8 | 1 0 | disp | | | | | | | |

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| Bcc | BHI d:8 | — | 4 2 | disp | | | | | | | | |
| | BHI d:16 | — | 5 8 | 2 0 | disp | | | | | | | |
| | BLS d:8 | — | 4 3 | disp | | | | | | | | |
| | BLS d:16 | — | 5 8 | 3 0 | disp | | | | | | | |
| | BCC d:8 (BHS d:8) | — | 4 4 | disp | | | | | | | | |
| | BCC d:16 (BHS d:16) | — | 5 8 | 4 0 | disp | | | | | | | |
| | BCS d:8 (BLO d:8) | — | 4 5 | disp | | | | | | | | |
| | BCS d:16 (BLO d:16) | — | 5 8 | 5 0 | disp | | | | | | | |
| | BNE d:8 | — | 4 6 | disp | | | | | | | | |
| | BNE d:16 | — | 5 8 | 6 0 | disp | | | | | | | |
| | BEQ d:8 | — | 4 7 | disp | | | | | | | | |
| | BEQ d:16 | — | 5 8 | 7 0 | disp | | | | | | | |
| | BVC d:8 | — | 4 8 | disp | | | | | | | | |
| | BVC d:16 | — | 5 8 | 8 0 | disp | | | | | | | |
| | BVS d:8 | — | 4 9 | disp | | | | | | | | |
| | BVS d:16 | — | 5 8 | 9 0 | disp | | | | | | | |
| | BPL d:8 | — | 4 A | disp | | | | | | | | |
| | BPL d:16 | — | 5 8 | A 0 | disp | | | | | | | |
| | BMI d:8 | — | 4 B | disp | | | | | | | | |
| | BMI d:16 | — | 5 8 | B 0 | disp | | | | | | | |
| | BGE d:8 | — | 4 C | disp | | | | | | | | |
| | BGE d:16 | — | 5 8 | C 0 | disp | | | | | | | |
| | BLT d:8 | — | 4 D | disp | | | | | | | | |
| | BLT d:16 | — | 5 8 | D 0 | disp | | | | | | | |
| | BGT d:8 | — | 4 E | disp | | | | | | | | |
| | BGT d:16 | — | 5 8 | E 0 | disp | | | | | | | |
| | BLE d:8 | — | 4 F | disp | | | | | | | | |
| | BLE d:16 | — | 5 8 | F 0 | disp | | | | | | | |

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BCLR | BCLR #xx:3,Rd | B | 7 | 2 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BCLR #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 7 | 2 | 0 IMM | 0 | | | | | | | | | | |
| | BCLR #xx:3,@aa:8 | B | 7 | F | abs | | 7 | 2 | 0 IMM | 0 | | | | | | | | | | |
| | BCLR #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 7 | 2 | 0 IMM | 0 | | | | | | |
| | BCLR #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 7 | 2 | 0 IMM | 0 | | |
| | BCLR Rn,Rd | B | 6 | 2 | rn | rd | | | | | | | | | | | | | | |
| | BCLR Rn,@ERd | B | 7 | D | 0 erd | 0 | 6 | 2 | rn | 0 | | | | | | | | | | |
| | BCLR Rn,@aa:8 | B | 7 | F | abs | | 6 | 2 | rn | 0 | | | | | | | | | | |
| | BCLR Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 2 | rn | 0 | | | | | | |
| | BCLR Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 2 | rn | 0 | | |
| BIAND | BIAND #xx:3,Rd | B | 7 | 6 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIAND #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 6 | 1 IMM | 0 | | | | | | | | | | |
| | BIAND #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 6 | 1 IMM | 0 | | | | | | | | | | |
| | BIAND #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 6 | 1 IMM | 0 | | | | | | |
| | BIAND #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 6 | 1 IMM | 0 | | |
| BILD | BILD #xx:3,Rd | B | 7 | 7 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BILD #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BILD #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BILD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 7 | 1 IMM | 0 | | | | | | |
| | BILD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 7 | 1 IMM | 0 | | |
| BIOR | BIOR #xx:3,Rd | B | 7 | 4 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIOR #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 4 | 1 IMM | 0 | | | | | | | | | | |
| | BIOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 4 | 1 IMM | 0 | | | | | | | | | | |
| | BIOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 4 | 1 IMM | 0 | | | | | | |
| | BIOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 4 | 1 IMM | 0 | | |

Table A-2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BIST | BIST #xx:3,Rd | B | 6 | 7 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIST #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 6 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BIST #xx:3,@aa:8 | B | 7 | F | abs | | 6 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BIST #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 7 | 1 IMM | 0 | | | | | | |
| | BIST #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 7 | 1 IMM | 0 | | |
| BIXOR | BIXOR #xx:3,Rd | B | 7 | 5 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIXOR #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 5 | 1 IMM | 0 | | | | | | | | | | |
| | BIXOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 5 | 1 IMM | 0 | | | | | | | | | | |
| | BIXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 5 | 1 IMM | 0 | | | | | | |
| | BIXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 5 | 1 IMM | 0 | | |
| BLD | BLD #xx:3,Rd | B | 7 | 7 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BLD #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 7 | 0 IMM | 0 | | | | | | | | | | |
| | BLD #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 7 | 0 IMM | 0 | | | | | | | | | | |
| | BLD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 7 | 0 IMM | 0 | | | | | | |
| | BLD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 7 | 0 IMM | 0 | | |
| BNOT | BNOT #xx:3,Rd | B | 7 | 1 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BNOT #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 7 | 1 | 0 IMM | 0 | | | | | | | | | | |
| | BNOT #xx:3,@aa:8 | B | 7 | F | abs | | 7 | 1 | 0 IMM | 0 | | | | | | | | | | |
| | BNOT #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 7 | 1 | 0 IMM | 0 | | | | | | |
| | BNOT #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 7 | 1 | 0 IMM | 0 | | |
| | BNOT Rn,Rd | B | 6 | 1 | rn | rd | | | | | | | | | | | | | | |
| | BNOT Rn,@ERd | B | 7 | D | 0 erd | 0 | 6 | 1 | rn | 0 | | | | | | | | | | |
| | BNOT Rn,@aa:8 | B | 7 | F | abs | | 6 | 1 | rn | 0 | | | | | | | | | | |
| | BNOT Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 1 | rn | 0 | | | | | | |
| | BNOT Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 1 | rn | 0 | | |

## Table A-2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | | | 2nd byte | | | | 3rd byte | | | | 4th byte | | | | 5th byte | | | | 6th byte | | | | 7th byte | | | | 8th byte | | | | 9th byte | | | | 10th byte | | | |
| BOR | BOR #xx:3,Rd | B | 7 | | 4 | | 0 | IMM | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BOR #xx:3,@ERd | B | 7 | | C | | 0 | erd | 0 | | 7 | | 4 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BOR #xx:3,@aa:8 | B | 7 | | E | | abs | | | | 7 | | 4 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BOR #xx:3,@aa:16 | B | 6 | | A | | 1 | | 0 | | abs | | | | | | | | 7 | | 4 | | 0 | IMM | 0 | | | | | | | | | | | |
| | BOR #xx:3,@aa:32 | B | 6 | | A | | 3 | | 0 | | abs | | | | | | | | | | | | | | | | 7 | | 4 | | 0 | IMM | 0 | | | |
| BSET | BSET #xx:3,Rd | B | 7 | | 0 | | 0 | IMM | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BSET #xx:3,@ERd | B | 7 | | D | | 0 | erd | 0 | | 7 | | 0 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BSET #xx:3,@aa:8 | B | 7 | | F | | abs | | | | 7 | | 0 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BSET #xx:3,@aa:16 | B | 6 | | A | | 1 | | 8 | | abs | | | | | | | | 7 | | 0 | | 0 | IMM | 0 | | | | | | | | | | | |
| | BSET #xx:3,@aa:32 | B | 6 | | A | | 3 | | 8 | | abs | | | | | | | | | | | | | | | | 7 | | 0 | | 0 | IMM | 0 | | | |
| | BSET Rn,Rd | B | 6 | | 0 | | rn | | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BSET Rn,@ERd | B | 7 | | D | | 0 | erd | 0 | | 6 | | 0 | | rn | | 0 | | | | | | | | | | | | | | | | | | | |
| | BSET Rn,@aa:8 | B | 7 | | F | | abs | | | | 6 | | 0 | | rn | | 0 | | | | | | | | | | | | | | | | | | | |
| | BSET Rn,@aa:16 | B | 6 | | A | | 1 | | 8 | | abs | | | | | | | | 6 | | 0 | | rn | | 0 | | | | | | | | | | | |
| | BSET Rn,@aa:32 | B | 6 | | A | | 3 | | 8 | | abs | | | | | | | | | | | | | | | | 6 | | 0 | | rn | | 0 | | | |
| BSR | BSR d:8 | — | 5 | | 5 | | disp | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BSR d:16 | — | 5 | | C | | 0 | | 0 | | disp | | | | | | | | | | | | | | | | | | | | | | | | | |
| BST | BST #xx:3,Rd | B | 6 | | 7 | | 0 | IMM | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BST #xx:3,@ERd | B | 7 | | D | | 0 | erd | 0 | | 6 | | 7 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BST #xx:3,@aa:8 | B | 7 | | F | | abs | | | | 6 | | 7 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BST #xx:3,@aa:16 | B | 6 | | A | | 1 | | 8 | | abs | | | | | | | | 6 | | 7 | | 0 | IMM | 0 | | | | | | | | | | | |
| | BST #xx:3,@aa:32 | B | 6 | | A | | 3 | | 8 | | abs | | | | | | | | | | | | | | | | 6 | | 7 | | 0 | IMM | 0 | | | |
| BTST | BTST #xx:3,Rd | B | 7 | | 3 | | 0 | IMM | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BTST #xx:3,@ERd | B | 7 | | C | | 0 | erd | 0 | | 7 | | 3 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BTST #xx:3,@aa:8 | B | 7 | | E | | abs | | | | 7 | | 3 | | 0 | IMM | 0 | | | | | | | | | | | | | | | | | | | |
| | BTST #xx:3,@aa:16 | B | 6 | | A | | 1 | | 0 | | abs | | | | | | | | 7 | | 3 | | 0 | IMM | 0 | | | | | | | | | | | |
| | BTST #xx:3,@aa:32 | B | 6 | | A | | 3 | | 0 | | abs | | | | | | | | | | | | | | | | 7 | | 3 | | 0 | IMM | 0 | | | |
| | BTST Rn,Rd | B | 6 | | 3 | | rn | | rd | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | BTST Rn,@ERd | B | 7 | | C | | 0 | erd | 0 | | 6 | | 3 | | rn | | 0 | | | | | | | | | | | | | | | | | | | |

Table A-2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BTST | BTST Rn,@aa:8 | B | 7 | E | abs | | 6 | 3 | rn | 0 | | | | | | | | | | |
| | BTST Rn,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 6 | 3 | rn | 0 | | | | | | |
| | BTST Rn,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 6 | 3 | rn | 0 | | |
| BXOR | BXOR #xx:3,Rd | B | 7 | 5 | 0 | IMM | rd | | | | | | | | | | | | | |
| | BXOR #xx:3,@ERd | B | 7 | C | 0 | erd | 0 | 7 | 5 | 0 | IMM | 0 | | | | | | | | |
| | BXOR #xx:3,@aa:8 | B | 7 | E | abs | | | 7 | 5 | 0 | IMM | 0 | | | | | | | | |
| | BXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 5 | 0 | IMM | 0 | | | | | |
| | BXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 5 | 0 | IMM | 0 | |
| CLRMAC | CLRMAC | — | 0 | 1 | A | 0 | | | | | | | | | | | | | | |
| CMP | CMP.B #xx:8,Rd | B | A | rd | IMM | | | | | | | | | | | | | | | |
| | CMP.B Rs,Rd | B | 1 | C | rs | rd | | | | | | | | | | | | | | |
| | CMP.W #xx:16,Rd | W | 7 | 9 | 2 | rd | IMM | | | | | | | | | | | | | |
| | CMP.W Rs,Rd | W | 1 | D | rs | rd | | | | | | | | | | | | | | |
| | CMP.L #xx:32,ERd | L | 7 | A | 2 | 0 | erd | IMM | | | | | | | | | | | | |
| | CMP.L ERs,ERd | L | 1 | F | 1 | ers | 0 | erd | | | | | | | | | | | | |
| DAA | DAA Rd | B | 0 | F | 0 | rd | | | | | | | | | | | | | | |
| DAS | DAS Rd | B | 1 | F | 0 | rd | | | | | | | | | | | | | | |
| DEC | DEC.B Rd | B | 1 | A | 0 | rd | | | | | | | | | | | | | | |
| | DEC.W #1,Rd | W | 1 | B | 5 | rd | | | | | | | | | | | | | | |
| | DEC.W #2,Rd | W | 1 | B | D | rd | | | | | | | | | | | | | | |
| | DEC.L #1,ERd | L | 1 | B | 7 | 0 | erd | | | | | | | | | | | | | |
| | DEC.L #2,ERd | L | 1 | B | F | 0 | erd | | | | | | | | | | | | | |
| DIVXS | DIVXS.B Rs,Rd | B | 0 | 1 | D | 0 | 5 | 1 | rs | rd | | | | | | | | | | |
| | DIVXS.W Rs,ERd | W | 0 | 1 | D | 0 | 5 | 3 | rs | 0 | erd | | | | | | | | | |
| DIVXU | DIVXU.B Rs,Rd | B | 5 | 1 | rs | rd | | | | | | | | | | | | | | |
| | DIVXU.W Rs,ERd | W | 5 | 3 | rs | 0 | erd | | | | | | | | | | | | | |
| EEPMOV | EEPMOV.B | — | 7 | B | 5 | C | 5 | 9 | 8 | F | | | | | | | | | | |
| | EEPMOV.W | — | 7 | B | D | 4 | 5 | 9 | 8 | F | | | | | | | | | | |

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | | 5th byte | | 6th byte | | 7th byte | | 8th byte | 9th byte | 10th byte |
| EXTS | EXTS.W Rd | W | 1 | 7 | D | rd | | | | | | | | | | | | | | |
| | EXTS.L ERd | L | 1 | 7 | F | 0 erd | | | | | | | | | | | | | | |
| EXTU | EXTU.W Rd | W | 1 | 7 | 5 | rd | | | | | | | | | | | | | | |
| | EXTU.L ERd | L | 1 | 7 | 7 | 0 erd | | | | | | | | | | | | | | |
| INC | INC.B Rd | B | 0 | A | 0 | rd | | | | | | | | | | | | | | |
| | INC.W #1,Rd | W | 0 | B | 5 | rd | | | | | | | | | | | | | | |
| | INC.W #2,Rd | W | 0 | B | D | rd | | | | | | | | | | | | | | |
| | INC.L #1,ERd | L | 0 | B | 7 | 0 erd | | | | | | | | | | | | | | |
| | INC.L #2,ERd | L | 0 | B | F | 0 erd | | | | | | | | | | | | | | |
| JMP | JMP @ERn | — | 5 | 9 | 0 ern | 0 | | | | | | | | | | | | | | |
| | JMP @aa:24 | — | 5 | A | abs | | | | | | | | | | | | | | | |
| | JMP @@aa:8 | — | 5 | B | abs | | | | | | | | | | | | | | | |
| JSR | JSR @ERn | — | 5 | D | 0 ern | 0 | | | | | | | | | | | | | | |
| | JSR @aa:24 | — | 5 | E | abs | | | | | | | | | | | | | | | |
| | JSR @@aa:8 | — | 5 | F | abs | | | | | | | | | | | | | | | |
| LDC | LDC #xx:8,CCR | B | 0 | 7 | IMM | | | | | | | | | | | | | | | |
| | LDC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0 | 7 | IMM | | | | | | | | | | | |
| | LDC Rs,CCR | B | 0 | 3 | 0 | rs | | | | | | | | | | | | | | |
| | LDC Rs,EXR | B | 0 | 3 | 1 | rs | | | | | | | | | | | | | | |
| | LDC @ERs,CCR | W | 0 | 1 | 4 | 0 | 6 | 9 | 0 ers | 0 | | | | | | | | | | |
| | LDC @ERs,EXR | W | 0 | 1 | 4 | 1 | 6 | 9 | 0 ers | 0 | | | | | | | | | | |
| | LDC @(d:16,ERs),CCR | W | 0 | 1 | 4 | 0 | 6 | F | 0 ers | 0 | | disp | | | | | | | | |
| | LDC @(d:16,ERs),EXR | W | 0 | 1 | 4 | 1 | 6 | F | 0 ers | 0 | | disp | | | | | | | | |
| | LDC @(d:32,ERs),CCR | W | 0 | 1 | 4 | 0 | 7 | 8 | 0 ers | 0 | 6 | B | 2 | 0 | disp | | | | | |
| | LDC @(d:32,ERs),EXR | W | 0 | 1 | 4 | 1 | 7 | 8 | 0 ers | 0 | 6 | B | 2 | 0 | disp | | | | | |
| | LDC @ERs+,CCR | W | 0 | 1 | 4 | 0 | 6 | D | 0 ers | 0 | | | | | | | | | | |
| | LDC @ERs+,EXR | W | 0 | 1 | 4 | 1 | 6 | D | 0 ers | 0 | | | | | | | | | | |
| | LDC @aa:16,CCR | W | 0 | 1 | 4 | 0 | 6 | B | 0 | 0 | | disp | | | | | | | | |
| | LDC @aa:16,EXR | W | 0 | 1 | 4 | 1 | 6 | B | 0 | 0 | | disp | | | | | | | | |

**Table A-2 Instruction Codes (cont)**

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| LDC | LDC @aa:32,CCR | W | 0 | 1 | 4 | 0 | 6 | B | 2 | 0 | abs | | | | | |
| | LDC @aa:32,EXR | W | 0 | 1 | 4 | 1 | 6 | B | 2 | 0 | abs | | | | | |
| LDM | LDM.L @SP+, (ERn-ERn+1) | L | 0 | 1 | 1 | 0 | 6 | D | 7 | 0 ern+1 | | | | | | |
| | LDM.L @SP+, (ERn-ERn+2) | L | 0 | 1 | 2 | 0 | 6 | D | 7 | 0 ern+2 | | | | | | |
| | LDM.L @SP+, (ERn-ERn+3) | L | 0 | 1 | 3 | 0 | 6 | D | 7 | 0 ern+3 | | | | | | |
| LDMAC | LDMAC ERs,MACH | L | 0 | 3 | 2 | 0 ers | | | | | | | | | | |
| | LDMAC ERs,MACL | L | 0 | 3 | 3 | 0 ers | | | | | | | | | | |
| MAC | MAC @ERn+,@ERm+ | — | 0 | 1 | 6 | 0 | 6 | D | 0 ern | 0 erm | | | | | | |
| MOV | MOV.B #xx:8,Rd | B | F | rd | IMM | | | | | | | | | | | |
| | MOV.B Rs,Rd | B | 0 | C | rs | rd | | | | | | | | | | |
| | MOV.B @ERs,Rd | B | 6 | 8 | 0 ers | rd | | | | | | | | | | |
| | MOV.B @(d:16,ERs),Rd | B | 6 | E | 0 ers | rd | disp | | | | | | | | | |
| | MOV.B @(d:32,ERs),Rd | B | 7 | 8 | 0 ers | 0 | 6 | A | 2 | rd | disp | | | | | |
| | MOV.B @ERs+,Rd | B | 6 | C | 0 ers | rd | | | | | | | | | | |
| | MOV.B @aa:8,Rd | B | 2 | rd | abs | | | | | | | | | | | |
| | MOV.B @aa:16,Rd | B | 6 | A | 0 | rd | abs | | | | | | | | | |
| | MOV.B @aa:32,Rd | B | 6 | A | 2 | rd | abs | | | | | | | | | |
| | MOV.B Rs,@ERd | B | 6 | 8 | 1 erd | rs | | | | | | | | | | |
| | MOV.B Rs,@(d:16,ERd) | B | 6 | E | 1 erd | rs | disp | | | | | | | | | |
| | MOV.B Rs,@(d:32,ERd) | B | 7 | 8 | 0 erd | 0 | 6 | A | A | rs | disp | | | | | |
| | MOV.B Rs,@-ERd | B | 6 | C | 1 erd | rs | | | | | | | | | | |
| | MOV.B Rs,@aa:8 | B | 3 | rs | abs | | | | | | | | | | | |
| | MOV.B Rs,@aa :16 | B | 6 | A | 8 | rs | abs | | | | | | | | | |
| | MOV.B Rs,@aa:32 | B | 6 | A | A | rs | abs | | | | | | | | | |
| | MOV.W #xx:16,Rd | W | 7 | 9 | 0 | rd | IMM | | | | | | | | | |
| | MOV.W Rs,Rd | W | 0 | D | rs | rd | | | | | | | | | | |
| | MOV.W @ERs,Rd | W | 6 | 9 | 0 ers | rd | | | | | | | | | | |
| | MOV.W @(d:16,ERs),Rd | W | 6 | F | 0 ers | rd | disp | | | | | | | | | |
| | MOV.W @(d:32,ERs),Rd | W | 7 | 8 | 0 ers | 0 | 6 | B | 2 | rd | disp | | | | | |

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| MOV | MOV.W @ERs+,Rd | W | 6 | D | 0 ers | rd | | | | | | | | | | | | | | |
| | MOV.W @aa:16,Rd | W | 6 | B | 0 | rd | abs | | | | | | | | | | | | | |
| | MOV.W @aa:32,Rd | W | 6 | B | 2 | rd | abs | | | | | | | | | | | | | |
| | MOV.W Rs,@ERd | W | 6 | 9 | 1 erd | rs | | | | | | | | | | | | | | |
| | MOV.W Rs,@(d:16,ERd) | W | 6 | F | 1 erd | rs | disp | | | | | | | | | | | | | |
| | MOV.W Rs,@(d:32,ERd) | W | 7 | 8 | 0 erd | 0 | 6 | B | A | rs | disp | | | | | | | | | |
| | MOV.W Rs,@-ERd | W | 6 | D | 1 erd | rs | | | | | | | | | | | | | | |
| | MOV.W Rs,@aa:16 | W | 6 | B | 8 | rs | abs | | | | | | | | | | | | | |
| | MOV.W Rs,@aa:32 | W | 6 | B | A | rs | abs | | | | | | | | | | | | | |
| | MOV.L #xx:32,Rd | L | 7 | A | 0 | 0 erd | IMM | | | | | | | | | | | | | |
| | MOV.L ERs,ERd | L | 0 | F | 1 ers | 0 erd | | | | | | | | | | | | | | |
| | MOV.L @ERs,ERd | L | 0 | 1 | 0 | 0 | 6 | 9 | 0 ers | 0 erd | | | | | | | | | | |
| | MOV.L @(d:16,ERs),ERd | L | 0 | 1 | 0 | 0 | 6 | F | 0 ers | 0 erd | disp | | | | | | | | | |
| | MOV.L @(d:32,ERs),ERd | L | 0 | 1 | 0 | 0 | 7 | 8 | 0 ers | 0 | 6 | B | 2 | 0 erd | disp | | | | | |
| | MOV.L @ERs+,ERd | L | 0 | 1 | 0 | 0 | 6 | D | 0 ers | 0 erd | | | | | | | | | | |
| | MOV.L @aa:16 ,ERd | L | 0 | 1 | 0 | 0 | 6 | B | 0 | 0 erd | abs | | | | | | | | | |
| | MOV.L @aa:32 ,ERd | L | 0 | 1 | 0 | 0 | 6 | B | 2 | 0 erd | abs | | | | | | | | | |
| | MOV.L ERs,@ERd | L | 0 | 1 | 0 | 0 | 6 | 9 | 1 erd | 0 ers | | | | | | | | | | |
| | MOV.L ERs,@(d:16,ERd) | L | 0 | 1 | 0 | 0 | 6 | F | 1 erd | 0 ers | disp | | | | | | | | | |
| | MOV.L ERs,@(d:32,ERd)* | L | 0 | 1 | 0 | 0 | 7 | 8 | 0 erd | 0 | 6 | B | A | 0 ers | disp | | | | | |
| | MOV.L ERs,@-ERd | L | 0 | 1 | 0 | 0 | 6 | D | 1 erd | 0 ers | | | | | | | | | | |
| | MOV.L ERs,@aa:16 | L | 0 | 1 | 0 | 0 | 6 | B | 8 | 0 ers | abs | | | | | | | | | |
| | MOV.L ERs,@aa:32 | L | 0 | 1 | 0 | 0 | 6 | B | A | 0 ers | abs | | | | | | | | | |
| MOVFPE | MOVFPE @aa:16,Rd | B | 6 | A | 4 | rd | abs | | | | | | | | | | | | | |
| MOVTPE | MOVTPE Rs,@aa:16 | B | 6 | A | C | rs | abs | | | | | | | | | | | | | |
| MULXS | MULXS.B Rs,Rd | B | 0 | 1 | C | 0 | 5 | 0 | rs | rd | | | | | | | | | | |
| | MULXS.W Rs,ERd | W | 0 | 1 | C | 0 | 5 | 2 | rs | 0 erd | | | | | | | | | | |
| MULXU | MULXU.B Rs,Rd | B | 5 | 0 | rs | rd | | | | | | | | | | | | | | |
| | MULXU.W Rs,ERd | W | 5 | 2 | rs | 0 erd | | | | | | | | | | | | | | |

Table A-2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| NEG | NEG.B Rd | B | 1 | 7 | 8 | rd | | | | | | | | | | |
| | NEG.W Rd | W | 1 | 7 | 9 | rd | | | | | | | | | | |
| | NEG.L ERd | L | 1 | 7 | B | 0 erd | | | | | | | | | | |
| NOP | NOP | — | 0 | 0 | 0 | 0 | | | | | | | | | | |
| NOT | NOT.B Rd | B | 1 | 7 | 0 | rd | | | | | | | | | | |
| | NOT.W Rd | W | 1 | 7 | 1 | rd | | | | | | | | | | |
| | NOT.L ERd | L | 1 | 7 | 3 | 0 erd | | | | | | | | | | |
| OR | OR.B #xx:8,Rd | B | C | rd | IMM | | | | | | | | | | | |
| | OR.B Rs,Rd | B | 1 | 4 | rs | rd | | | | | | | | | | |
| | OR.W #xx:16,Rd | W | 7 | 9 | 4 | rd | IMM | | | | | | | | | |
| | OR.W Rs,Rd | W | 6 | 4 | rs | rd | | | | | | | | | | |
| | OR.L #xx:32,ERd | L | 7 | A | 4 | 0 erd | IMM | | | | | | | | | |
| | OR.L ERs,ERd | L | 0 | 1 | F | 0 | 6 | 4 | 0 ers | 0 erd | | | | | | |
| ORC | ORC #xx:8,CCR | B | 0 | 4 | IMM | | | | | | | | | | | |
| | ORC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0 | 4 | IMM | | | | | | | |
| POP | POP.W Rn | W | 6 | D | 7 | rn | | | | | | | | | | |
| | POP.L ERn | L | 0 | 1 | 0 | 0 | 6 | D | 7 | 0 ern | | | | | | |
| PUSH | PUSH.W Rn | W | 6 | D | F | rn | | | | | | | | | | |
| | PUSH.L ERn | L | 0 | 1 | 0 | 0 | 6 | D | F | 0 ern | | | | | | |
| ROTL | ROTL.B Rd | B | 1 | 2 | 8 | rd | | | | | | | | | | |
| | ROTL.B #2, Rd | B | 1 | 2 | C | rd | | | | | | | | | | |
| | ROTL.W Rd | W | 1 | 2 | 9 | rd | | | | | | | | | | |
| | ROTL.W #2, Rd | W | 1 | 2 | D | rd | | | | | | | | | | |
| | ROTL.L ERd | L | 1 | 2 | B | 0 erd | | | | | | | | | | |
| | ROTL.L #2, ERd | L | 1 | 2 | F | 0 erd | | | | | | | | | | |

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte | | | | | | |
| ROTR | ROTR.B Rd | B | 1 | 3 | 8 | rd | | | | | | | | | | | | | | |
| | ROTR.B #2, Rd | B | 1 | 3 | C | rd | | | | | | | | | | | | | | |
| | ROTR.W Rd | W | 1 | 3 | 9 | rd | | | | | | | | | | | | | | |
| | ROTR.W #2, Rd | W | 1 | 3 | D | rd | | | | | | | | | | | | | | |
| | ROTR.L ERd | L | 1 | 3 | B | 0 erd | | | | | | | | | | | | | | |
| | ROTR.L #2, ERd | L | 1 | 3 | F | 0 erd | | | | | | | | | | | | | | |
| ROTXL | ROTXL.B Rd | B | 1 | 2 | 0 | rd | | | | | | | | | | | | | | |
| | ROTXL.B #2, Rd | B | 1 | 2 | 4 | rd | | | | | | | | | | | | | | |
| | ROTXL.W Rd | W | 1 | 2 | 1 | rd | | | | | | | | | | | | | | |
| | ROTXL.W #2, Rd | W | 1 | 2 | 5 | rd | | | | | | | | | | | | | | |
| | ROTXL.L ERd | L | 1 | 2 | 3 | 0 erd | | | | | | | | | | | | | | |
| | ROTXL.L #2, ERd | L | 1 | 2 | 7 | 0 erd | | | | | | | | | | | | | | |
| ROTXR | ROTXR.B Rd | B | 1 | 3 | 0 | rd | | | | | | | | | | | | | | |
| | ROTXR.B #2, Rd | B | 1 | 3 | 4 | rd | | | | | | | | | | | | | | |
| | ROTXR.W Rd | W | 1 | 3 | 1 | rd | | | | | | | | | | | | | | |
| | ROTXR.W #2, Rd | W | 1 | 3 | 5 | rd | | | | | | | | | | | | | | |
| | ROTXR.L ERd | L | 1 | 3 | 3 | 0 erd | | | | | | | | | | | | | | |
| | ROTXR.L #2, ERd | L | 1 | 3 | 7 | 0 erd | | | | | | | | | | | | | | |
| RTE | RTE | — | 5 | 6 | 7 | 0 | | | | | | | | | | | | | | |
| RTS | RTS | — | 5 | 4 | 7 | 0 | | | | | | | | | | | | | | |
| SHAL | SHAL.B Rd | B | 1 | 0 | 8 | rd | | | | | | | | | | | | | | |
| | SHAL.B #2, Rd | B | 1 | 0 | C | rd | | | | | | | | | | | | | | |
| | SHAL.W Rd | W | 1 | 0 | 9 | rd | | | | | | | | | | | | | | |
| | SHAL.W #2, Rd | W | 1 | 0 | D | rd | | | | | | | | | | | | | | |
| | SHAL.L ERd | L | 1 | 0 | B | 0 erd | | | | | | | | | | | | | | |
| | SHAL.L #2, ERd | L | 1 | 0 | F | 0 erd | | | | | | | | | | | | | | |

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | | | 5th byte | | 6th byte | | 7th byte | 8th byte | 9th byte | 10th byte |
| SHAR | SHAR.B Rd | B | 1 | 1 | 8 | rd | | | | | | | | | | | | | | |
| | SHAR.B #2, Rd | B | 1 | 1 | C | rd | | | | | | | | | | | | | | |
| | SHAR.W Rd | W | 1 | 1 | 9 | rd | | | | | | | | | | | | | | |
| | SHAR.W #2, Rd | W | 1 | 1 | D | rd | | | | | | | | | | | | | | |
| | SHAR.L ERd | L | 1 | 1 | B | 0 erd | | | | | | | | | | | | | | |
| | SHAR.L #2, ERd | L | 1 | 1 | F | 0 erd | | | | | | | | | | | | | | |
| SHLL | SHLL.B Rd | B | 1 | 0 | 0 | rd | | | | | | | | | | | | | | |
| | SHLL.B #2, Rd | B | 1 | 0 | 4 | rd | | | | | | | | | | | | | | |
| | SHLL.W Rd | W | 1 | 0 | 1 | rd | | | | | | | | | | | | | | |
| | SHLL.W #2, Rd | W | 1 | 0 | 5 | rd | | | | | | | | | | | | | | |
| | SHLL.L ERd | L | 1 | 0 | 3 | 0 erd | | | | | | | | | | | | | | |
| | SHLL.L #2, ERd | L | 1 | 0 | 7 | 0 erd | | | | | | | | | | | | | | |
| SHLR | SHLR.B Rd | B | 1 | 1 | 0 | rd | | | | | | | | | | | | | | |
| | SHLR.B #2, Rd | B | 1 | 1 | 4 | rd | | | | | | | | | | | | | | |
| | SHLR.W Rd | W | 1 | 1 | 1 | rd | | | | | | | | | | | | | | |
| | SHLR.W #2, Rd | W | 1 | 1 | 5 | rd | | | | | | | | | | | | | | |
| | SHLR.L ERd | L | 1 | 1 | 3 | 0 erd | | | | | | | | | | | | | | |
| | SHLR.L #2, ERd | L | 1 | 1 | 7 | 0 erd | | | | | | | | | | | | | | |
| SLEEP | SLEEP | — | 0 | 1 | 8 | 0 | | | | | | | | | | | | | | |
| STC | STC.B CCR,Rd | B | 0 | 2 | 0 | rd | | | | | | | | | | | | | | |
| | STC.B EXR,Rd | B | 0 | 2 | 1 | rd | | | | | | | | | | | | | | |
| | STC.W CCR,@ERd | W | 0 | 1 | 4 | 0 | 6 | 9 | 1 erd | 0 | | | | | | | | | | |
| | STC.W EXR,@ERd | W | 0 | 1 | 4 | 1 | 6 | 9 | 1 erd | 0 | | | | | | | | | | |
| | STC.W CCR,@(d:16,ERd) | W | 0 | 1 | 4 | 0 | 6 | F | 1 erd | 0 | disp | | | | | | | | | |
| | STC.W EXR,@(d:16,ERd) | W | 0 | 1 | 4 | 1 | 6 | F | 1 erd | 0 | disp | | | | | | | | | |
| | STC.W CCR,@(d:32,ERd) | W | 0 | 1 | 4 | 0 | 7 | 8 | 0 erd | 0 | 6 | B | A | 0 | disp | | | | | |
| | STC.W EXR,@(d:32,ERd) | W | 0 | 1 | 4 | 1 | 7 | 8 | 0 erd | 0 | 6 | B | A | 0 | disp | | | | | |
| | STC.W CCR,@-ERd | W | 0 | 1 | 4 | 0 | 6 | D | 1 erd | 0 | | | | | | | | | | |
| | STC.W EXR,@-ERd | W | 0 | 1 | 4 | 1 | 6 | D | 1 erd | 0 | | | | | | | | | | |

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| STC | STC.W CCR,@aa:16 | W | 0 | 1 | 4 | 0 | 6 | B | 8 | 0 | abs | | | | | |
| | STC.W EXR,@aa:16 | W | 0 | 1 | 4 | 1 | 6 | B | 8 | 0 | abs | | | | | |
| | STC.W CCR,@aa:32 | W | 0 | 1 | 4 | 0 | 6 | B | A | 0 | abs | | | | | |
| | STC.W EXR,@aa:32 | W | 0 | 1 | 4 | 1 | 6 | B | A | 0 | abs | | | | | |
| STM | STM.L(ERn-ERn+1), @-SP | L | 0 | 1 | 1 | 0 | 6 | D | F | 0 ern | | | | | | |
| | STM.L (ERn-ERn+2), @-SP | L | 0 | 1 | 2 | 0 | 6 | D | F | 0 ern | | | | | | |
| | STM.L (ERn-ERn+3), @-SP | L | 0 | 1 | 3 | 0 | 6 | D | F | 0 ern | | | | | | |
| STMAC | STMAC MACH,ERd | L | 0 | 2 | 2 | 0 ers | | | | | | | | | | |
| | STMAC MACL,ERd | L | 0 | 2 | 3 | 0 ers | | | | | | | | | | |
| SUB | SUB.B Rs,Rd | B | 1 | 8 | rs | rd | | | | | | | | | | |
| | SUB.W #xx:16,Rd | W | 7 | 9 | 3 | rd | IMM | | | | | | | | | |
| | SUB.W Rs,Rd | W | 1 | 9 | rs | rd | | | | | | | | | | |
| | SUB.L #xx:32,ERd | L | 7 | A | 3 | 0 erd | IMM | | | | | | | | | |
| | SUB.L ERs,ERd | L | 1 | A | 1 ers | 0 erd | | | | | | | | | | |
| SUBS | SUBS #1,ERd | L | 1 | B | 0 | 0 erd | | | | | | | | | | |
| | SUBS #2,ERd | L | 1 | B | 8 | 0 erd | | | | | | | | | | |
| | SUBS #4,ERd | L | 1 | B | 9 | 0 erd | | | | | | | | | | |
| SUBX | SUBX #xx:8,Rd | B | B | rd | IMM | | | | | | | | | | | |
| | SUBX Rs,Rd | B | 1 | E | rs | rd | | | | | | | | | | |
| TAS | TAS @ERd | B | 0 | 1 | E | 0 | 7 | B | 0 erd | C | | | | | | |
| TRAPA | TRAPA #x:2 | — | 5 | 7 | 00 IMM | 0 | | | | | | | | | | |
| XOR | XOR.B #xx:8,Rd | B | D | rd | IMM | | | | | | | | | | | |
| | XOR.B Rs,Rd | B | 1 | 5 | rs | rd | | | | | | | | | | |
| | XOR.W #xx:16,Rd | W | 7 | 9 | 5 | rd | IMM | | | | | | | | | |
| | XOR.W Rs,Rd | W | 6 | 5 | rs | rd | | | | | | | | | | |
| | XOR.L #xx:32,ERd | L | 7 | A | 5 | 0 erd | IMM | | | | | | | | | |
| | XOR.L ERs,ERd | L | 0 | 1 | F | 0 | 6 | 5 | 0 ers | 0 erd | | | | | | |

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| XORC | XORC #xx:8,CCR | B | 0 | 5 | IMM | | | | | | | | | | |
| | XORC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0 | 5 | IMM | | | | | | |

Note: * Bit 7 of the 4th byte of the MOV.L ERs, @(d:32,ERd) instruction can be either 1 or 0.

**Legend**

| | |
|---|---|
| IMM: | Immediate data (2, 3, 8, 16, or 32 bits) |
| abs: | Absolute address (8, 16, 24, or 32 bits) |
| disp: | Displacement (8, 16, or 32 bits) |
| rs, rd, rn: | Register field (4 bits specifying an 8-bit or 16-bit register. The symbols rs, rd, and rn correspond to operand symbols Rs, Rd,and Rn.) |
| ers, erd, ern, erm: | Register field (3 bits specifying an address register or 32-bit register. The symbols ers, erd, ern, and erm correspond to operand symbols ERs, ERd, ERn, and ERm.) |

The register fields specify general registers as follows.

| Address Register 32-Bit Register | | 16-Bit Register | | 8-Bit Register | |
|---|---|---|---|---|---|
| Register Field | General Register | Register Field | General Register | Register Field | General Register |
| 000 | ER0 | 0000 | R0 | 0000 | R0H |
| 001 | ER1 | 0001 | R1 | 0001 | R1H |
| • | • | • | • | • | • |
| • | • | • | • | • | • |
| • | • | • | • | • | • |
| 111 | ER7 | 0111 | R7 | 0111 | R7H |
| | | 1000 | E0 | 1000 | R0L |
| | | 1001 | E1 | 1001 | R1L |
| | | • | • | • | • |
| | | • | • | • | • |
| | | • | • | • | • |
| | | 1111 | E7 | 1111 | R7L |

## A.3 Operation Code Map

Table A-3 shows the operation code map.

**Table A-3 Operation Code Map (1)**

## Table A-3 Operation Code Map (2)

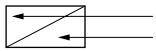
Instruction code

| 1st byte | | 2nd byte | |
|---|---|---|---|
| AH | AL | BH | BL |

| AH AL \ BH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | MOV | LDM / STM | | | LDC / STC | | MAC | | SLEEP | | CLRMAC | | Table A.3(3) | Table A.3(3) | TAS | Table A.3(3) |
| 0A | INC | | | | | | | | ADD | | | | | | | |
| 0B | ADDS | | | | | INC | | INC | ADDS | | | | | INC | | INC |
| 0F | DAA | | | | | | | | MOV | | | | | | | |
| 10 | SHLL | | | SHLL | | | | SHLL | SHAL | | | SHAL | | | | SHAL |
| 11 | SHLR | | | SHLR | | | | SHLR | SHAR | | | SHAR | | | | SHAR |
| 12 | ROTXL | | | ROTXL | | | | ROTXL | ROTL | | | ROTL | | | | ROTL |
| 13 | ROTXR | | | ROTXR | | | | ROTXR | ROTR | | | ROTR | | | | ROTR |
| 17 | NOT | | | NOT | | EXTU | | EXTU | NEG | | | NEG | | EXTS | | EXTS |
| 1A | DEC | | | | | | | | SUB | | | | | | | |
| 1B | SUBS | | | | | DEC | | DEC | SUBS | | | | | DEC | | DEC |
| 1F | DAS | | | | | | | | CMP | | | | | | | |
| 58 | BRA | BRN | BHI | BLS | BCC | BCS | BNE | BEQ | BVC | BVS | BPL | BMI | BGE | BLT | BGT | BLE |
| 6A | MOV | Table A.3(4) | MOV | Table A.3(4) | MOVFPE | | | | MOV | | MOV | | MOVTPE | | | |
| 79 | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | |
| 7A | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | |

# Table A-3 Operation Code Map (3)

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | |
|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL |



Instruction when most significant bit of DH is 0.
Instruction when most significant bit of DH is 1.

| CL / AH AL BH BL CH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 01C05 | MULXS | | MULXS | | | | | | | | | | | | | |
| 01D05 | | DIVXS | | DIVXS | | | | | | | | | | | | |
| 01F06 | | | | | OR | XOR | AND | | | | | | | | | |
| 7Cr06 *1 | | | | BTST | | | | | | | | | | | | |
| 7Cr07 *1 | | | | BTST | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 7Dr06 *1 | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 7Dr07 *1 | BSET | BNOT | BCLR | | | | | | | | | | | | | |
| 7Eaa6 *2 | | | | BTST | | | | | | | | | | | | |
| 7Eaa7 *2 | | | | BTST | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 7Faa6 *2 | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 7Faa7 *2 | BSET | BNOT | BCLR | | | | | | | | | | | | | |

Notes: 1. r is the register specification field.
2. aa is the absolute address specification.

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL |

Instruction when most significant bit of FH is 0.
Instruction when most significant bit of FH is 1.

| EL / AHALBHBLCHCLDHDLEH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6A10aaaa6* | | | | BTST | | | | | | | | | | | | |
| 6A10aaaa7* | | | | | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 6A18aaaa6* | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 6A18aaaa7* | | | | | | | | | | | | | | | | |

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL | GH | GL | HH | HL |

Instruction when most significant bit of HH is 0.
Instruction when most significant bit of HH is 1.

| GL / AHALBHBL ... FHFLGH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6A30aaaaaaaa6* | | | | BTST | | | | | | | | | | | | |
| 6A30aaaaaaaa7* | | | | | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 6A38aaaaaaaa6* | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 6A38aaaaaaaa7* | | | | | | | | | | | | | | | | |

Note: * aa is the absolute address specification.

## A.4 Number of States Required for Instruction Execution

The tables in this section can be used to calculate the number of states required for instruction execution by the CPU. Table A-5 indicates the number of instruction fetch, data read/write, and other cycles occurring in each instruction. Table A-4 indicates the number of states required for each cycle. The number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

**Examples:** Advanced mode, program code and stack located in external memory, on-chip supporting modules accessed in two states with 8-bit bus width, external devices accessed in three states with one wait state and 16-bit bus width.

1. BSET #0, @FFFFC7:8

   From table A-5:

   $I = L = 2, \quad J = K = M = N = 0$

   From table A-4:

   $S_I = 4, \quad S_L = 2$

   Number of states required for execution = $2 \times 4 + 2 \times 2 = 12$

2. JSR @@30

   From table A-5:

   $I = J = K = 2, \quad L = M = N = 0$

   From table A-4:

   $S_I = S_J = S_K = 4$

   Number of states required for execution = $2 \times 4 + 2 \times 4 + 2 \times 4 = 24$

**Table A-4 Number of States per Cycle**

| | | Access Conditions | | | | | |
|---|---|---|---|---|---|---|---|
| | | On-Chip Supporting Module | | External Device | | | |
| | | | | 8-Bit Bus | | 16-Bit Bus | |
| Cycle | | On-Chip Memory | 8-Bit Bus | 16-Bit Bus | 2-State Access | 3-State Access | 2-State Access | 3-State Access |
| Instruction fetch | $S_I$ | 1 | 4 | 2 | 4 | 6 + 2m | 2 | 3 + m |
| Branch address read | $S_J$ | | | | | | | |
| Stack operation | $S_K$ | | | | | | | |
| Byte data access | $S_L$ | | 2 | | 2 | 3 + m | | |
| Word data access | $S_M$ | | 4 | | 4 | 6 + 2m | | |
| Internal operation | $S_N$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Legend

m: Number of wait states inserted into external device access

## Table A-5 Number of Cycles in Instruction Execution

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| ADD | ADD.B #xx:8,Rd | 1 | | | | | |
| | ADD.B Rs,Rd | 1 | | | | | |
| | ADD.W #xx:16,Rd | 2 | | | | | |
| | ADD.W Rs,Rd | 1 | | | | | |
| | ADD.L #xx:32,ERd | 3 | | | | | |
| | ADD.L ERs,ERd | 1 | | | | | |
| ADDS | ADDS #1/2/4,ERd | 1 | | | | | |
| ADDX | ADDX #xx:8,Rd | 1 | | | | | |
| | ADDX Rs,Rd | 1 | | | | | |
| AND | AND.B #xx:8,Rd | 1 | | | | | |
| | AND.B Rs,Rd | 1 | | | | | |
| | AND.W #xx:16,Rd | 2 | | | | | |
| | AND.W Rs,Rd | 1 | | | | | |
| | AND.L #xx:32,ERd | 3 | | | | | |
| | AND.L ERs,ERd | 2 | | | | | |
| ANDC | ANDC #xx:8,CCR | 1 | | | | | |
| | ANDC #xx:8,EXR | 2 | | | | | |
| BAND | BAND #xx:3,Rd | 1 | | | | | |
| | BAND #xx:3,@ERd | 2 | | | 1 | | |
| | BAND #xx:3,@aa:8 | 2 | | | 1 | | |
| | BAND #xx:3,@aa:16 | 3 | | | 1 | | |
| | BAND #xx:3,@aa:32 | 4 | | | 1 | | |
| Bcc | BRA d:8 (BT d:8) | 2 | | | | | |
| | BRN d:8 (BF d:8) | 2 | | | | | |
| | BHI d:8 | 2 | | | | | |
| | BLS d:8 | 2 | | | | | |
| | BCC d:8 (BHS d:8) | 2 | | | | | |
| | BCS d:8 (BLO d:8) | 2 | | | | | |
| | BNE d:8 | 2 | | | | | |
| | BEQ d:8 | 2 | | | | | |
| | BVC d:8 | 2 | | | | | |
| | BVS d:8 | 2 | | | | | |
| | BPL d:8 | 2 | | | | | |
| | BMI d:8 | 2 | | | | | |
| | BGE d:8 | 2 | | | | | |
| | BLT d:8 | 2 | | | | | |
| | BGT d:8 | 2 | | | | | |
| | BLE d:8 | 2 | | | | | |
| | BRA d:16 (BT d:16) | 2 | | | | | 1 |
| | BRN d:16 (BF d:16) | 2 | | | | | 1 |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| Bcc | BHI d:16 | 2 | | | | | 1 |
| | BLS d:16 | 2 | | | | | 1 |
| | BCC d:16 (BHS d:16) | 2 | | | | | 1 |
| | BCS d:16 (BLO d:16) | 2 | | | | | 1 |
| | BNE d:16 | 2 | | | | | 1 |
| | BEQ d:16 | 2 | | | | | 1 |
| | BVC d:16 | 2 | | | | | 1 |
| | BVS d:16 | 2 | | | | | 1 |
| | BPL d:16 | 2 | | | | | 1 |
| | BMI d:16 | 2 | | | | | 1 |
| | BGE d:16 | 2 | | | | | 1 |
| | BLT d:16 | 2 | | | | | 1 |
| | BGT d:16 | 2 | | | | | 1 |
| | BLE d:16 | 2 | | | | | 1 |
| BCLR | BCLR #xx:3,Rd | 1 | | | | | |
| | BCLR #xx:3,@ERd | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:8 | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:16 | 3 | | | 2 | | |
| | BCLR #xx:3,@aa:32 | 4 | | | 2 | | |
| | BCLR Rn,Rd | 1 | | | | | |
| | BCLR Rn,@ERd | 2 | | | 2 | | |
| | BCLR Rn,@aa:8 | 2 | | | 2 | | |
| | BCLR Rn,@aa:16 | 3 | | | 2 | | |
| | BCLR Rn,@aa:32 | 4 | | | 2 | | |
| BIAND | BIAND #xx:3,Rd | 1 | | | | | |
| | BIAND #xx:3,@ERd | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIAND #xx:3,@aa:32 | 4 | | | 1 | | |
| BILD | BILD #xx:3,Rd | 1 | | | | | |
| | BILD #xx:3,@ERd | 2 | | | 1 | | |
| | BILD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BILD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BILD #xx:3,@aa:32 | 4 | | | 1 | | |
| BIOR | BIOR #xx:8,Rd | 1 | | | | | |
| | BIOR #xx:8,@ERd | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:8 | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:16 | 3 | | | 1 | | |
| | BIOR #xx:8,@aa:32 | 4 | | | 1 | | |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| BIST | BIST #xx:3,Rd | 1 | | | | | |
| | BIST #xx:3,@ERd | 2 | | | 2 | | |
| | BIST #xx:3,@aa:8 | 2 | | | 2 | | |
| | BIST #xx:3,@aa:16 | 3 | | | 2 | | |
| | BIST #xx:3,@aa:32 | 4 | | | 2 | | |
| BIXOR | BIXOR #xx:3,Rd | 1 | | | | | |
| | BIXOR #xx:3,@ERd | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIXOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BLD | BLD #xx:3,Rd | 1 | | | | | |
| | BLD #xx:3,@ERd | 2 | | | 1 | | |
| | BLD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BLD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BLD #xx:3,@aa:32 | 4 | | | 1 | | |
| BNOT | BNOT #xx:3,Rd | 1 | | | | | |
| | BNOT #xx:3,@ERd | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:8 | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:16 | 3 | | | 2 | | |
| | BNOT #xx:3,@aa:32 | 4 | | | 2 | | |
| | BNOT Rn,Rd | 1 | | | | | |
| | BNOT Rn,@ERd | 2 | | | 2 | | |
| | BNOT Rn,@aa:8 | 2 | | | 2 | | |
| | BNOT Rn,@aa:16 | 3 | | | 2 | | |
| | BNOT Rn,@aa:32 | 4 | | | 2 | | |
| BOR | BOR #xx:3,Rd | 1 | | | | | |
| | BOR #xx:3,@ERd | 2 | | | 1 | | |
| | BOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BSET | BSET #xx:3,Rd | 1 | | | | | |
| | BSET #xx:3,@ERd | 2 | | | 2 | | |
| | BSET #xx:3,@aa:8 | 2 | | | 2 | | |
| | BSET #xx:3,@aa:16 | 3 | | | 2 | | |
| | BSET #xx:3,@aa:32 | 4 | | | 2 | | |
| | BSET Rn,Rd | 1 | | | | | |
| | BSET Rn,@ERd | 2 | | | 2 | | |
| | BSET Rn,@aa:8 | 2 | | | 2 | | |
| | BSET Rn,@aa:16 | 3 | | | 2 | | |
| | BSET Rn,@aa:32 | 4 | | | 2 | | |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|---|
| BSR | BSR d:8 | Normal | 2 | | 1 | | | |
| | | Advanced | 2 | | 2 | | | |
| | BSR d:16 | Normal | 2 | | 1 | | | 1 |
| | | Advanced | 2 | | 2 | | | 1 |
| BST | BST #xx:3,Rd | | 1 | | | | | |
| | BST #xx:3,@ERd | | 2 | | | 2 | | |
| | BST #xx:3,@aa:8 | | 2 | | | 2 | | |
| | BST #xx:3,@aa:16 | | 3 | | | 2 | | |
| | BST #xx:3,@aa:32 | | 4 | | | 2 | | |
| BTST | BTST #xx:3,Rd | | 1 | | | | | |
| | BTST #xx:3,@ERd | | 2 | | | 1 | | |
| | BTST #xx:3,@aa:8 | | 2 | | | 1 | | |
| | BTST #xx:3,@aa:16 | | 3 | | | 1 | | |
| | BTST #xx:3,@aa:32 | | 4 | | | 1 | | |
| | BTST Rn,Rd | | 1 | | | | | |
| | BTST Rn,@ERd | | 2 | | | 1 | | |
| | BTST Rn,@aa:8 | | 2 | | | 1 | | |
| | BTST Rn,@aa:16 | | 3 | | | 1 | | |
| | BTST Rn,@aa:32 | | 4 | | | 1 | | |
| BXOR | BXOR #xx:3,Rd | | 1 | | | | | |
| | BXOR #xx:3,@ERd | | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:8 | | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:16 | | 3 | | | 1 | | |
| | BXOR #xx:3,@aa:32 | | 4 | | | 1 | | |
| CLRMAC | CLRMAC | | 1 | | | | | 1*[3] |
| CMP | CMP.B #xx:8,Rd | | 1 | | | | | |
| | CMP.B Rs,Rd | | 1 | | | | | |
| | CMP.W #xx:16,Rd | | 2 | | | | | |
| | CMP.W Rs,Rd | | 1 | | | | | |
| | CMP.L #xx:32,ERd | | 3 | | | | | |
| | CMP.L ERs,ERd | | 1 | | | | | |
| DAA | DAA Rd | | 1 | | | | | |
| DAS | DAS Rd | | 1 | | | | | |
| DEC | DEC.B Rd | | 1 | | | | | |
| | DEC.W #1/2,Rd | | 1 | | | | | |
| | DEC.L #1/2,ERd | | 1 | | | | | |
| DIVXS | DIVXS.B Rs,Rd | | 2 | | | | | 11 |
| | DIVXS.W Rs,ERd | | 2 | | | | | 19 |
| DIVXU | DIVXU.B Rs,Rd | | 1 | | | | | 11 |
| | DIVXU.W Rs,ERd | | 1 | | | | | 19 |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|---|
| EEPMOV | EEPMOV.B | | 2 | | | 2n + 2*[1] | | |
| | EEPMOV.W | | 2 | | | 2n + 2*[1] | | |
| EXTS | EXTS.W Rd | | 1 | | | | | |
| | EXTS.L ERd | | 1 | | | | | |
| EXTU | EXTU.W Rd | | 1 | | | | | |
| | EXTU.L ERd | | 1 | | | | | |
| INC | INC.B Rd | | 1 | | | | | |
| | INC.W #1/2,Rd | | 1 | | | | | |
| | INC.L #1/2,ERd | | 1 | | | | | |
| JMP | JMP @ERn | | 2 | | | | | |
| | JMP @aa:24 | | 2 | | | | | 1 |
| | JMP @@aa:8 | Normal | 2 | 1 | | | | 1 |
| | | Advanced | 2 | 2 | | | | 1 |
| JSR | JSR @ERn | Normal | 2 | | 1 | | | |
| | | Advanced | 2 | | 2 | | | |
| | JSR @aa:24 | Normal | 2 | | 1 | | | 1 |
| | | Advanced | 2 | | 2 | | | 1 |
| | JSR @@aa:8 | Normal | 2 | 1 | 1 | | | |
| | | Advanced | 2 | 2 | 2 | | | |
| LDC | LDC #xx:8,CCR | | 1 | | | | | |
| | LDC #xx:8,EXR | | 2 | | | | | |
| | LDC Rs,CCR | | 1 | | | | | |
| | LDC Rs,EXR | | 1 | | | | | |
| | LDC @ERs,CCR | | 2 | | | | 1 | |
| | LDC @ERs,EXR | | 2 | | | | 1 | |
| | LDC @(d:16,ERs),CCR | | 3 | | | | 1 | |
| | LDC @(d:16,ERs),EXR | | 3 | | | | 1 | |
| | LDC @(d:32,ERs),CCR | | 5 | | | | 1 | |
| | LDC @(d:32,ERs),EXR | | 5 | | | | 1 | |
| | LDC @ERs+,CCR | | 2 | | | | 1 | 1 |
| | LDC @ERs+,EXR | | 2 | | | | 1 | 1 |
| | LDC @aa:16,CCR | | 3 | | | | 1 | |
| | LDC @aa:16,EXR | | 3 | | | | 1 | |
| | LDC @aa:32,CCR | | 4 | | | | 1 | |
| | LDC @aa:32,EXR | | 4 | | | | 1 | |
| LDM | LDM.L @SP+, (ERn-ERn+1) | | 2 | | 4 | | | 1 |
| | LDM.L @SP+, (ERn-ERn+2) | | 2 | | 6 | | | 1 |
| | LDM.L @SP+, (ERn-ERn+3) | | 2 | | 8 | | | 1 |
| LDMAC | LDMAC ERs,MACH | | 1 | | | | | 1*[3] |
| | LDMAC ERs,MACL | | 1 | | | | | 1*[3] |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| MAC | MAC @ERn+,@ERm+ | 2 | | | | 2 | |
| MOV | MOV.B #xx:8,Rd | 1 | | | | | |
| | MOV.B Rs,Rd | 1 | | | | | |
| | MOV.B @ERs,Rd | 1 | | | 1 | | |
| | MOV.B @(d:16,ERs),Rd | 2 | | | 1 | | |
| | MOV.B @(d:32,ERs),Rd | 4 | | | 1 | | |
| | MOV.B @ERs+,Rd | 1 | | | 1 | | 1 |
| | MOV.B @aa:8,Rd | 1 | | | 1 | | |
| | MOV.B @aa:16,Rd | 2 | | | 1 | | |
| | MOV.B @aa:32,Rd | 3 | | | 1 | | |
| | MOV.B Rs,@ERd | 1 | | | 1 | | |
| | MOV.B Rs,@(d:16,ERd) | 2 | | | 1 | | |
| | MOV.B Rs,@(d:32,ERd) | 4 | | | 1 | | |
| | MOV.B Rs,@-ERd | 1 | | | 1 | | 1 |
| | MOV.B Rs,@aa:8 | 1 | | | 1 | | |
| | MOV.B Rs,@aa:16 | 2 | | | 1 | | |
| | MOV.B Rs,@aa:32 | 3 | | | 1 | | |
| | MOV.W #xx:16,Rd | 2 | | | | | |
| | MOV.W Rs,Rd | 1 | | | | | |
| | MOV.W @ERs,Rd | 1 | | | | 1 | |
| | MOV.W @(d:16,ERs),Rd | 2 | | | | 1 | |
| | MOV.W @(d:32,ERs),Rd | 4 | | | | 1 | |
| | MOV.W @ERs+,Rd | 1 | | | | 1 | 1 |
| | MOV.W @aa:16,Rd | 2 | | | | 1 | |
| | MOV.W @aa:32,Rd | 3 | | | | 1 | |
| | MOV.W Rs,@ERd | 1 | | | | 1 | |
| | MOV.W Rs,@(d:16,ERd) | 2 | | | | 1 | |
| | MOV.W Rs,@(d:32,ERd) | 4 | | | | 1 | |
| | MOV.W Rs,@-ERd | 1 | | | | 1 | 1 |
| | MOV.W Rs,@aa:16 | 2 | | | | 1 | |
| | MOV.W Rs,@aa:32 | 3 | | | | 1 | |
| | MOV.L #xx:32,ERd | 3 | | | | | |
| | MOV.L ERs,ERd | 1 | | | | | |
| | MOV.L @ERs,ERd | 2 | | | | 2 | |
| | MOV.L @(d:16,ERs),ERd | 3 | | | | 2 | |
| | MOV.L @(d:32,ERs),ERd | 5 | | | | 2 | |
| | MOV.L @ERs+,ERd | 2 | | | | 2 | 1 |
| | MOV.L @aa:16,ERd | 3 | | | | 2 | |
| | MOV.L @aa:32,ERd | 4 | | | | 2 | |
| | MOV.L ERs,@ERd | 2 | | | | 2 | |
| | MOV.L ERs,@(d:16,ERd) | 3 | | | | 2 | |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| MOV | MOV.L ERs,@(d:32,ERd) | 5 | | | | 2 | |
| | MOV.L ERs,@-ERd | 2 | | | | 2 | 1 |
| | MOV.L ERs,@aa:16 | 3 | | | | 2 | |
| | MOV.L ERs,@aa:32 | 4 | | | | 2 | |
| MOVFPE | MOVFPE @:aa:16,Rd | Can not be used in the H8S/2655 Series. | | | | | |
| MOVTPE | MOVTPE Rs,@:aa:16 | | | | | | |
| MULXS | MULXS.B Rs,Rd | 2 | | | | | 2*[3] |
| | MULXS.W Rs,ERd | 2 | | | | | 3*[3] |
| MULXU | MULXU.B Rs,Rd | 1 | | | | | 2*[3] |
| | MULXU.W Rs,ERd | 1 | | | | | 3*[3] |
| NEG | NEG.B Rd | 1 | | | | | |
| | NEG.W Rd | 1 | | | | | |
| | NEG.L ERd | 1 | | | | | |
| NOP | NOP | 1 | | | | | |
| NOT | NOT.B Rd | 1 | | | | | |
| | NOT.W Rd | 1 | | | | | |
| | NOT.L ERd | 1 | | | | | |
| OR | OR.B #xx:8,Rd | 1 | | | | | |
| | OR.B Rs,Rd | 1 | | | | | |
| | OR.W #xx:16,Rd | 2 | | | | | |
| | OR.W Rs,Rd | 1 | | | | | |
| | OR.L #xx:32,ERd | 3 | | | | | |
| | OR.L ERs,ERd | 2 | | | | | |
| ORC | ORC #xx:8,CCR | 1 | | | | | |
| | ORC #xx:8,EXR | 2 | | | | | |
| POP | POP.W Rn | 1 | | | | 1 | 1 |
| | POP.L ERn | 2 | | | | 2 | 1 |
| PUSH | PUSH.W Rn | 1 | | | | 1 | 1 |
| | PUSH.L ERn | 2 | | | | 2 | 1 |
| ROTL | ROTL.B Rd | 1 | | | | | |
| | ROTL.B #2,Rd | 1 | | | | | |
| | ROTL.W Rd | 1 | | | | | |
| | ROTL.W #2,Rd | 1 | | | | | |
| | ROTL.L ERd | 1 | | | | | |
| | ROTL.L #2,ERd | 1 | | | | | |
| ROTR | ROTR.B Rd | 1 | | | | | |
| | ROTR.B #2,Rd | 1 | | | | | |
| | ROTR.W Rd | 1 | | | | | |
| | ROTR.W #2,Rd | 1 | | | | | |
| | ROTR.L ERd | 1 | | | | | |
| | ROTR.L #2,ERd | 1 | | | | | |

## Table A-5 Number of Cycles in Instruction Execution (cont)

| Instruction | Mnemonic | | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|---|
| ROTXL | ROTXL.B Rd | | 1 | | | | | |
| | ROTXL.B #2,Rd | | 1 | | | | | |
| | ROTXL.W Rd | | 1 | | | | | |
| | ROTXL.W #2,Rd | | 1 | | | | | |
| | ROTXL.L ERd | | 1 | | | | | |
| | ROTXL.L #2,ERd | | 1 | | | | | |
| ROTXR | ROTXR.B Rd | | 1 | | | | | |
| | ROTXR.B #2,Rd | | 1 | | | | | |
| | ROTXR.W Rd | | 1 | | | | | |
| | ROTXR.W #2,Rd | | 1 | | | | | |
| | ROTXR.L ERd | | 1 | | | | | |
| | ROTXR.L #2,ERd | | 1 | | | | | |
| RTE | RTE | | 2 | | 2/3*[1] | | | 1 |
| RTS | RTS | Normal | 2 | | 1 | | | 1 |
| | | Advanced | 2 | | 2 | | | 1 |
| SHAL | SHAL.B Rd | | 1 | | | | | |
| | SHAL.B #2,Rd | | 1 | | | | | |
| | SHAL.W Rd | | 1 | | | | | |
| | SHAL.W #2,Rd | | 1 | | | | | |
| | SHAL.L ERd | | 1 | | | | | |
| | SHAL.L #2,ERd | | 1 | | | | | |
| SHAR | SHAR.B Rd | | 1 | | | | | |
| | SHAR.B #2,Rd | | 1 | | | | | |
| | SHAR.W Rd | | 1 | | | | | |
| | SHAR.W #2,Rd | | 1 | | | | | |
| | SHAR.L ERd | | 1 | | | | | |
| | SHAR.L #2,ERd | | 1 | | | | | |
| SHLL | SHLL.B Rd | | 1 | | | | | |
| | SHLL.B #2,Rd | | 1 | | | | | |
| | SHLL.W Rd | | 1 | | | | | |
| | SHLL.W #2,Rd | | 1 | | | | | |
| | SHLL.L ERd | | 1 | | | | | |
| | SHLL.L #2,ERd | | 1 | | | | | |
| SHLR | SHLR.B Rd | | 1 | | | | | |
| | SHLR.B #2,Rd | | 1 | | | | | |
| | SHLR.W Rd | | 1 | | | | | |
| | SHLR.W #2,Rd | | 1 | | | | | |
| | SHLR.L ERd | | 1 | | | | | |
| | SHLR.L #2,ERd | | 1 | | | | | |
| SLEEP | SLEEP | | 1 | | | | | 1 |

## Table A-5 Number of Cycles in Instruction Execution (cont)

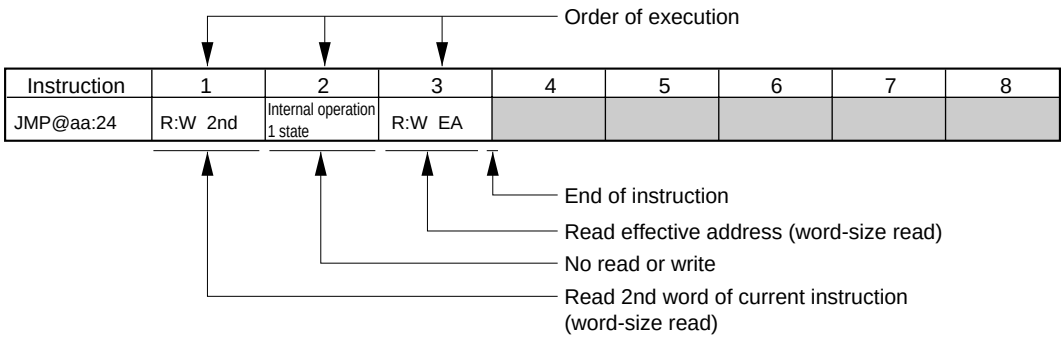
| Instruction | Mnemonic | | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|---|
| STC | STC.B CCR,Rd | | 1 | | | | | |
| | STC.B EXR,Rd | | 1 | | | | | |
| | STC.W CCR,@ERd | | 2 | | | | 1 | |
| | STC.W EXR,@ERd | | 2 | | | | 1 | |
| | STC.W CCR,@(d:16,ERd) | | 3 | | | | 1 | |
| | STC.W EXR,@(d:16,ERd) | | 3 | | | | 1 | |
| | STC.W CCR,@(d:32,ERd) | | 5 | | | | 1 | |
| | STC.W EXR,@(d:32,ERd) | | 5 | | | | 1 | |
| | STC.W CCR,@-ERd | | 2 | | | | 1 | 1 |
| | STC.W EXR,@-ERd | | 2 | | | | 1 | 1 |
| | STC.W CCR,@aa:16 | | 3 | | | | 1 | |
| | STC.W EXR,@aa:16 | | 3 | | | | 1 | |
| | STC.W CCR,@aa:32 | | 4 | | | | 1 | |
| | STC.W EXR,@aa:32 | | 4 | | | | 1 | |
| STM | STM.L (ERn-ERn+1),@-SP | | 2 | | 4 | | | 1 |
| | STM.L (ERn-ERn+2),@-SP | | 2 | | 6 | | | 1 |
| | STM.L (ERn-ERn+3),@-SP | | 2 | | 8 | | | 1 |
| STMAC*3 | STMAC MACH,ERd | | 1 | | | | | *3 |
| | STMAC MACL,ERd | | 1 | | | | | *3 |
| SUB | SUB.B Rs,Rd | | 1 | | | | | |
| | SUB.W #xx:16,Rd | | 2 | | | | | |
| | SUB.W Rs,Rd | | 1 | | | | | |
| | SUB.L #xx:32,ERd | | 3 | | | | | |
| | SUB.L ERs,ERd | | 1 | | | | | |
| SUBS | SUBS #1/2/4,ERd | | 1 | | | | | |
| SUBX | SUBX #xx:8,Rd | | 1 | | | | | |
| | SUBX Rs,Rd | | 1 | | | | | |
| TAS | TAS @ERd | | 2 | | | 2 | | |
| TRAPA | TRAPA #x:2 | Normal | 2 | 1 | 2/3*1 | | | 2 |
| | | Advanced | 2 | 2 | 2/3*1 | | | 2 |
| XOR | XOR.B #xx:8,Rd | | 1 | | | | | |
| | XOR.B Rs,Rd | | 1 | | | | | |
| | XOR.W #xx:16,Rd | | 2 | | | | | |
| | XOR.W Rs,Rd | | 1 | | | | | |
| | XOR.L #xx:32,ERd | | 3 | | | | | |
| | XOR.L ERs,ERd | | 2 | | | | | |
| XORC | XORC #xx:8,CCR | | 1 | | | | | |
| | XORC #xx:8,EXR | | 2 | | | | | |

Notes: 1. 2 when EXR is invalid, 3 when EXR is valid.
2. 5 for concatenated execution, 4 otherwise.
3. An internal operation may require between 0 and 3 additional states, depending on the preceding instruction.

## A.5 Bus States During Instruction Execution

Table A-6 indicates the types of cycles that occur during instruction execution by the CPU. See table A-4 for the number of states per cycle.

**How to Read the Table:**




**Legend**

| | |
|---|---|
| R:B | Byte-size read |
| R:W | Word-size read |
| W:B | Byte-size write |
| W:W | Word-size write |
| :M | Transfer of the bus is not performed immediately after this cycle |
| 2nd | Address of 2nd word (3rd and 4th bytes) |
| 3rd | Address of 3rd word (5th and 6th bytes) |
| 4th | Address of 4th word (7th and 8th bytes) |
| 5th | Address of 5th word (9th and 10th bytes) |
| NEXT | Address of next instruction |
| EA | Effective address |
| VEC | Vector address |

Figure A-1 shows timing waveforms for the address bus and the $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$ signals during execution of the above instruction with an 8-bit bus, using three-state access with no wait states.

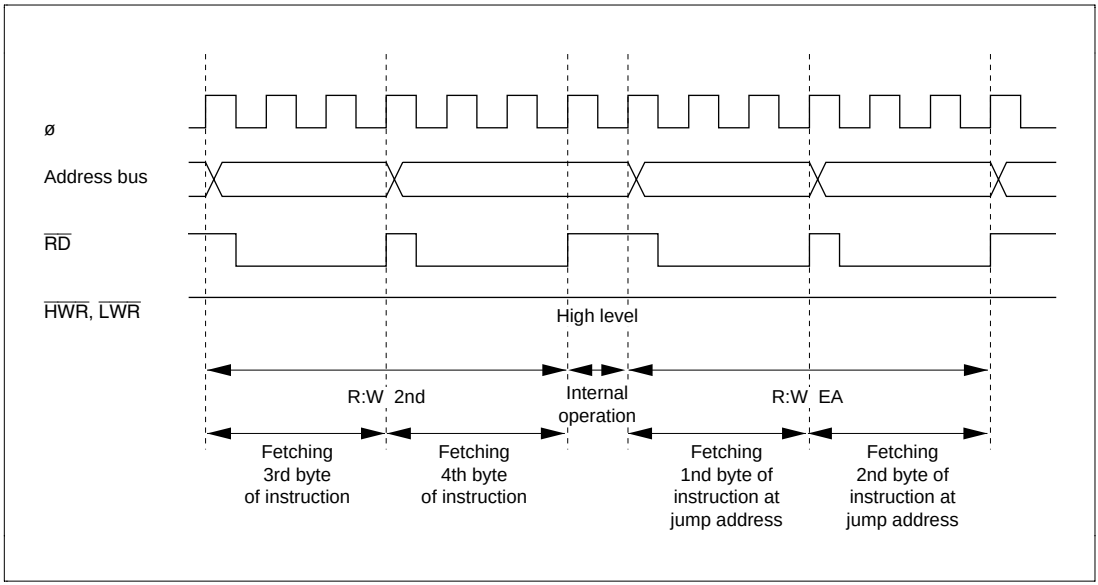



**Figure A-1 Address Bus, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$ Timing (8-Bit Bus, Three-State Access, No Wait States)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| ADD.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADD.B Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| ADD.W Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| ADD.L ERs,ERd | R:W NEXT | | | | | | | | |
| ADDS #1/2/4,ERd | R:W NEXT | | | | | | | | |
| ADDX #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADDX Rs,Rd | R:W NEXT | | | | | | | | |
| AND.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| AND.B Rs,Rd | R:W NEXT | | | | | | | | |
| AND.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| AND.W Rs,Rd | R:W NEXT | | | | | | | | |
| AND.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| AND.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ANDC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ANDC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| BAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BAND #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BRA d:8 (BT d:8) | R:W NEXT | R:W EA | | | | | | | |
| BRN d:8 (BF d:8) | R:W NEXT | R:W EA | | | | | | | |
| BHI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BCC d:8 (BHS d:8) | R:W NEXT | R:W EA | | | | | | | |
| BCS d:8 (BLO d:8) | R:W NEXT | R:W EA | | | | | | | |
| BNE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BEQ d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVC d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BPL d:8 | R:W NEXT | R:W EA | | | | | | | |
| BMI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLT d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGT d:8 | R:W NEXT | R:W EA | | | | | | | |

**Table A-6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BLE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BRA d:16 (BT d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BRN d:16 (BF d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BHI d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLS d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCC d:16 (BHS d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCS d:16 (BLO d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BNE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BEQ d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BVC d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BVS d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BPL d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BMI d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BGE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLT d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BGT d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCLR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BCLR #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BCLR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BCLR Rn,Rd | R:W NEXT | | | | | | | | |
| BCLR Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BCLR Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIAND #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BILD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BILD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BILD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIST #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BIST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIXOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIXOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIXOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BLD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BLD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BLD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BNOT #xx:3,Rd | R:W NEXT | | | | | | | | |

| Instruction | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| BNOT #xx:3,@ERd | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:8 | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:16 | | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BNOT #xx:3,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BNOT Rn,Rd | | R:W NEXT | | | | | | | | |
| BNOT Rn,@ERd | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT Rn,@aa:8 | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT Rn,@aa:16 | | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BNOT Rn,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BOR #xx:3,Rd | | R:W NEXT | | | | | | | | |
| BOR #xx:3,@ERd | | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:8 | | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:16 | | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BOR #xx:3,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BSET #xx:3,Rd | | R:W NEXT | | | | | | | | |
| BSET #xx:3,@ERd | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:8 | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:16 | | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET #xx:3,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSET Rn,Rd | | R:W NEXT | | | | | | | | |
| BSET Rn,@ERd | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:8 | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:16 | | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET Rn,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSR d:8 | Normal | R:W NEXT | R:W EA | W:W stack | | | | | | |
| | Advanced | R:W NEXT | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | | |
| BSR d:16 | Normal | R:W 2nd | Internal operation, 1 state | R:W EA | W:W stack | | | | | |
| | Advanced | R:W 2nd | Internal operation, 1 state | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | |
| BST #xx:3,Rd | | R:W NEXT | | | | | | | | |
| BST #xx:3,@ERd | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:8 | | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:16 | | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BST #xx:3,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BTST #xx:3,Rd | | R:W NEXT | | | | | | | | |
| BTST #xx:3,@ERd | | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BTST #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BTST Rn,Rd | R:W NEXT | | | | | | | | |
| BTST Rn,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BXOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BXOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BXOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BXOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BXOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| CLRMAC | R:W NEXT | Internal operation, 1 state | | | | | | | |
| CMP.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| CMP.B Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| CMP.W Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| CMP.L ERs,ERd | R:W NEXT | | | | | | | | |
| DAA Rd | R:W NEXT | | | | | | | | |
| DAS Rd | R:W NEXT | | | | | | | | |
| DEC.B Rd | R:W NEXT | | | | | | | | |
| DEC.W #1/2,Rd | R:W NEXT | | | | | | | | |
| DEC.L #1/2,ERd | R:W NEXT | | | | | | | | |
| DIVXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation, 11 states | | | | | | |
| DIVXS.W Rs,ERd | R:W 2nd | R:W NEXT | Internal operation, 19 states | | | | | | |
| DIVXU.B Rs,Rd | R:W NEXT | Internal operation, 11 states | | | | | | | |
| DIVXU.W Rs,ERd | R:W NEXT | Internal operation, 19 states | | | | | | | |
| EEPMOV.B | R:W 2nd | R:B EAs*1 | R:B EAd*1 | R:B EAs*2 | W:B EAd*2 | R:W NEXT | | | |
| EEPMOV.W | R:W 2nd | R:B EAs*1 | R:B EAd*1 | R:B EAs*2 | W:B EAd*2 | R:W NEXT | | | |
| EXTS.W Rd | R:W NEXT | | | ← Repeated n times*2 → | | | | | |
| EXTS.L ERd | R:W NEXT | | | | | | | | |
| EXTU.W Rd | R:W NEXT | | | | | | | | |
| EXTU.L ERd | R:W NEXT | | | | | | | | |
| INC.B Rd | R:W NEXT | | | | | | | | |

| Instruction | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| INC.W #1/2,Rd | | R:W NEXT | | | | | | | | |
| INC.L #1/2,ERd | | R:W NEXT | | | | | | | | |
| JMP @ERn | | R:W NEXT | R:W EA | | | | | | | |
| JMP @aa:24 | | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| JMP @@aa:8 | Normal | R:W NEXT | R:W aa:8 | Internal operation, 1 state | R:W EA | | | | | |
| | Advanced | R:W NEXT | R:W:M aa:8 | R:W aa:8 | Internal operation, 1 state | R:W EA | | | | |
| JSR @ERn | Normal | R:W NEXT | R:W EA | W:W stack | | | | | | |
| | Advanced | R:W NEXT | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | | |
| JSR @aa:24 | Normal | R:W 2nd | Internal operation, 1 state | R:W EA | W:W stack | | | | | |
| | Advanced | R:W 2nd | Internal operation, 1 state | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | |
| JSR @@aa:8 | Normal | R:W NEXT | R:W aa:8 | W:W stack | R:W EA | | | | | |
| | Advanced | R:W NEXT | R:W:M aa:8 | R:W aa:8 | W:W:M stack (H) | W:W stack (L) | R:W EA | | | |
| LDC #xx:8,CCR | | R:W NEXT | | | | | | | | |
| LDC #xx:8,EXR | | R:W 2nd | R:W NEXT | | | | | | | |
| LDC Rs,CCR | | R:W NEXT | | | | | | | | |
| LDC Rs,EXR | | R:W NEXT | | | | | | | | |
| LDC @ERs,CCR | | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LDC @ERs,EXR | | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LDC @(d:16,ERs),CCR | | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @(d:16,ERs),EXR | | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @(d:32,ERs),CCR | | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LDC @(d:32,ERs),EXR | | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LDC @ERs+,CCR | | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | |
| LDC @ERs+,EXR | | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | |
| LDC @aa:16,CCR | | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @aa:16,EXR | | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @aa:32,CCR | | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LDC @aa:32,EXR | | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LDM.L @SP+, (ERn–ERn+1) | | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| LDM.L @SP+,(ERn–ERn+2) | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |
| LDM.L @SP+,(ERn–ERn+3) | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |
| LDMAC ERs,MACH | R:W NEXT | Internal operation, 1 state | | ← Repeated n times*3 → | | | | | |
| LDMAC ERs,MACL | R:W NEXT | Internal operation, 1 state | | | | | | | |
| MAC @ERn+,@ERm+ | R:W 2nd | R:W NEXT | R:W EAn | R:W EAm | | | | | |
| MOV.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| MOV.B Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.B @ERs,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:B EA | | | | |
| MOV.B @ERs+,Rd | R:W NEXT | Internal operation, 1 state | R:B EA | | | | | | |
| MOV.B @aa:8,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @aa:16,Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.B Rs,@ERd | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:B EA | | | | |
| MOV.B Rs,@–ERd | R:W NEXT | Internal operation, 1 state | W:B EA | | | | | | |
| MOV.B Rs,@aa:8 | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@aa:16 | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:B EA | | | | | |
| MOV.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| MOV.W Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.W @ERs,Rd | R:W NEXT | R:W EA | | | | | | | |
| MOV.W @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| MOV.W @ERs+, Rd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | | |
| MOV.W @aa:16,Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.W Rs,@ERd | R:W NEXT | W:W EA | | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| MOV.W Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:E 4th | R:W NEXT | W:W EA | | | | |
| MOV.W Rs,@–ERd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | | |
| MOV.W Rs,@aa:16 | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| MOV.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| MOV.L ERs,ERd | R:W NEXT | | | | | | | | |
| MOV.L @ERs,ERd | R:W 2nd | R:W:M NEXT | R:W:M EA | R:W EA+2 | | | | | |
| MOV.L @(d:16,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @(d:32,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W:M 4th | R:W 5th | R:W NEXT | R:W:M EA | R:W EA+2 | | |
| MOV.L @ERs+,ERd | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:16,ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:32,ERd | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | R:W:M EA | R:W EA+2 | | | |
| MOV.L ERs,@ERd | R:W 2nd | R:W:M NEXT | W:W:M EA | W:W EA+2 | | | | | |
| MOV.L ERs,@(d:16,ERd) | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@(d:32,ERd) | R:W 2nd | R:W:M 3rd | R:W:M 4th | R:W 5th | R:W NEXT | W:W:M EA | W:W EA+2 | | |
| MOV.L ERs,@–ERd | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@aa:16 | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@aa:32 | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | W:W:M EA | W:W EA+2 | | | |
| MOVFPE @aa:16,Rd*4 | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOVTPE Rs,@aa:16*4 | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MULXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation, 2 states | | | | | | |
| MULXS.W Rs,ERd | R:W 2nd | R:W NEXT | Internal operation, 3 states | | | | | | |
| MULXU.B Rs,Rd | R:W NEXT | Internal operation, 2 states | | | | | | | |
| MULXU.W Rs,ERd | R:W NEXT | Internal operation, 3 states | | | | | | | |
| NEG.B Rd | R:W NEXT | | | | | | | | |
| NEG.W Rd | R:W NEXT | | | | | | | | |
| NEG.L ERd | R:W NEXT | | | | | | | | |
| NOP | R:W NEXT | | | | | | | | |
| NOT.B Rd | R:W NEXT | | | | | | | | |
| NOT.W Rd | R:W NEXT | | | | | | | | |
| NOT.L ERd | R:W NEXT | | | | | | | | |
| OR.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| OR.B Rs,Rd | R:W NEXT | | | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| OR.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| OR.W Rs,Rd | R:W NEXT | | | | | | | | |
| OR.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| OR.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ORC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ORC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| POP.W Rn | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | | |
| POP.L ERn | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M EA | R:W EA+2 | | | | |
| PUSH.W Rn | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | | |
| PUSH.L ERn | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M EA | W:W EA+2 | | | | |
| ROTL.B Rd | R:W NEXT | | | | | | | | |
| ROTL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.W Rd | R:W NEXT | | | | | | | | |
| ROTL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.L ERd | R:W NEXT | | | | | | | | |
| ROTL.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTR.B Rd | R:W NEXT | | | | | | | | |
| ROTR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.W Rd | R:W NEXT | | | | | | | | |
| ROTR.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.L ERd | R:W NEXT | | | | | | | | |
| ROTR.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXL.B Rd | R:W NEXT | | | | | | | | |
| ROTXL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.W Rd | R:W NEXT | | | | | | | | |
| ROTXL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.L ERd | R:W NEXT | | | | | | | | |
| ROTXL.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXR.B Rd | R:W NEXT | | | | | | | | |
| ROTXR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXR.W Rd | R:W NEXT | | | | | | | | |
| ROTXR.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTXR.L ERd | R:W NEXT | | | | | | | | |

**Table A-6 Instruction Execution Cycles (cont)**

| Instruction | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| ROTXR.L #2,ERd | | R:W NEXT | | | | | | | | |
| RTE | | R:W NEXT | R:W stack (EXR) | R:W stack (H) | R:W stack (L) | Internal operation, 1 state | R:W*5 | | | |
| RTS | Normal | R:W NEXT | R:W stack | Internal operation, 1 state | R:W*5 | | | | | |
| | Advanced | R:W NEXT | R:W:M stack (H) | R:W stack (L) | Internal operation, 1 state | R:W*5 | | | | |
| SHAL.B Rd | | R:W NEXT | | | | | | | | |
| SHAL.B #2,Rd | | R:W NEXT | | | | | | | | |
| SHAL.W Rd | | R:W NEXT | | | | | | | | |
| SHAL.W #2,Rd | | R:W NEXT | | | | | | | | |
| SHAL.L ERd | | R:W NEXT | | | | | | | | |
| SHAL.L #2,ERd | | R:W NEXT | | | | | | | | |
| SHAR.B Rd | | R:W NEXT | | | | | | | | |
| SHAR.B #2,Rd | | R:W NEXT | | | | | | | | |
| SHAR.W Rd | | R:W NEXT | | | | | | | | |
| SHAR.W #2,Rd | | R:W NEXT | | | | | | | | |
| SHAR.L ERd | | R:W NEXT | | | | | | | | |
| SHAR.L #2,ERd | | R:W NEXT | | | | | | | | |
| SHLL.B Rd | | R:W NEXT | | | | | | | | |
| SHLL.B #2,Rd | | R:W NEXT | | | | | | | | |
| SHLL.W Rd | | R:W NEXT | | | | | | | | |
| SHLL.W #2,Rd | | R:W NEXT | | | | | | | | |
| SHLL.L ERd | | R:W NEXT | | | | | | | | |
| SHLL.L #2,ERd | | R:W NEXT | | | | | | | | |
| SHLR.B Rd | | R:W NEXT | | | | | | | | |
| SHLR.B #2,Rd | | R:W NEXT | | | | | | | | |
| SHLR.W Rd | | R:W NEXT | | | | | | | | |
| SHLR.W #2,Rd | | R:W NEXT | | | | | | | | |
| SHLR.L ERd | | R:W NEXT | | | | | | | | |
| SHLR.L #2,ERd | | R:W NEXT | | | | | | | | |
| SLEEP | | R:W NEXT | Internal operation:M | | | | | | | |
| STC CCR,Rd | | R:W NEXT | | | | | | | | |
| STC EXR,Rd | | R:W NEXT | | | | | | | | |
| STC CCR,@ERd | | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC EXR,@ERd | | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC CCR,@(d:16,ERd) | | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |

**Table A-6 Instruction Execution Cycles (cont)**

| Instruction | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| STC EXR,@(d:16,ERd) | | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@(d:32,ERd) | | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC EXR,@(d:32,ERd) | | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC CCR,@–ERd | | R:W 2nd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | |
| STC EXR,@–ERd | | R:W 2nd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | |
| STC CCR,@aa:16 | | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC EXR,@aa:16 | | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STC EXR,@aa:32 | | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STM.L(ERn–ERn+1),@–SP | | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STM.L(ERn–ERn+2),@–SP | | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STM.L(ERn–ERn+3),@–SP | | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STMAC MACH,ERd | | R:W NEXT | | | ←Repeated n times*3→ | | | | | |
| STMAC MACL,ERd | | R:W NEXT | | | | | | | | |
| SUB.B Rs,Rd | | R:W NEXT | | | | | | | | |
| SUB.W #xx:16,Rd | | R:W 2nd | R:W NEXT | | | | | | | |
| SUB.W Rs,Rd | | R:W NEXT | | | | | | | | |
| SUB.L #xx:32,ERd | | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| SUB.L ERs,ERd | | R:W NEXT | | | | | | | | |
| SUBS #1/2/4,ERd | | R:W NEXT | | | | | | | | |
| SUBX #xx:8,Rd | | R:W NEXT | | | | | | | | |
| SUBX Rs,Rd | | R:W NEXT | | | | | | | | |
| TAS @ERd | | R:W 2nd | R:W NEXT | R:B:M EA | W:B EA | | | | | |
| TRAPA #x:2 | Normal | R:W NEXT | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W VEC | Internal operation, 1 state | R:W*8 | |
| | Advanced | R:W NEXT | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W:M VEC | R:W VEC+2 | Internal operation, 1 state | R:W*8 |
| XOR.B #xx8,Rd | | R:W NEXT | | | | | | | | |
| XOR.B Rs,Rd | | R:W NEXT | | | | | | | | |
| XOR.W #xx:16,Rd | | R:W 2nd | R:W NEXT | | | | | | | |
| XOR.W Rs,Rd | | R:W NEXT | | | | | | | | |
| XOR.L #xx:32,ERd | | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |

| Instruction | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| XOR.L ERs,ERd | | R:W 2nd | R:W NEXT | | | | | | | |
| XORC #xx:8,CCR | | R:W NEXT | | | | | | | | |
| XORC #xx:8,EXR | | R:W 2nd | R:W NEXT | | | | | | | |
| Reset exception handling | Normal | R:W VEC | Internal operation, 1 state | R:W*6 | | | | | | |
| | Advanced | R:W VEC | R:W VEC+2 | Internal operation, 1 state | R:W*6 | | | | | |
| Interrupt exception handling | Normal | R:W*7 | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W VEC | Internal operation, 1 state | R:W*8 | |
| | Advanced | R:W*7 | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W:M VEC | R:W VEC+2 | Internal operation, 1 state | R:W*8 |

Notes:
1. EAs is the contents of ER5. EAd is the contents of ER6.
2. EAs is the contents of ER5. EAd is the contents of ER6. Both registers are incremented by 1 after execution of the instruction. n is the initial value of R4L or R4. If n = 0, these bus cycles are not executed.
3. Repeated two times to save or restore two registers, three times for three registers, or four times for four registers.
4. Can not be used in the H8S/2655 Series.
5. Start address after return.
6. Start address of the program.
7. Prefetch address, equal to two plus the PC value pushed onto the stack. In recovery from sleep mode or software standby mode the read operation is replaced by an internal operation.
8. Start address of the interrupt-handling routine.

## A.6 Condition Code Modification

This section indicates the effect of each CPU instruction on the condition code. The notation used in the table is defined below.

m = { 31 for longword operands; 15 for word operands; 7 for byte operands

| | |
|---|---|
| Si | The i-th bit of the source operand |
| Di | The i-th bit of the destination operand |
| Ri | The i-th bit of the result |
| Dn | The specified bit in the destination operand |
| — | Not affected |
| ↕ | Modified according to the result of the instruction (see definition) |
| 0 | Always cleared to 0 |
| 1 | Always set to 1 |
| * | Undetermined (no guaranteed value) |
| Z' | Z flag before instruction execution |
| C' | C flag before instruction execution |

**Table A-7 Condition Code Modification**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| ADD | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · Dm–4 + Dm–4 · $\overline{Rm–4}$ + Sm–4 · $\overline{Rm–4}$<br>N = Rm<br>Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$<br>V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm<br>C = Sm · Dm + Dm · $\overline{Rm}$ + Sm · $\overline{Rm}$ |
| ADDS | — | — | — | — | — | |
| ADDX | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · Dm–4 + Dm–4 · $\overline{Rm–4}$ + Sm–4 · $\overline{Rm–4}$<br>N = Rm<br>Z = Z' · $\overline{Rm}$ · ····· · $\overline{R0}$<br>V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm<br>C = Sm · Dm + Dm · $\overline{Rm}$ + Sm · $\overline{Rm}$ |
| AND | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| ANDC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result.<br>No flags change when the operand is EXR. |
| BAND | — | — | — | — | ↕ | C = C' · Dn |
| Bcc | — | — | — | — | — | |
| BCLR | — | — | — | — | — | |
| BIAND | — | — | — | — | ↕ | C = C' · $\overline{Dn}$ |
| BILD | — | — | — | — | ↕ | C = $\overline{Dn}$ |
| BIOR | — | — | — | — | ↕ | C = C' + $\overline{Dn}$ |
| BIST | — | — | — | — | — | |
| BIXOR | — | — | — | — | ↕ | C = C' · Dn + $\overline{C'}$ · $\overline{Dn}$ |
| BLD | — | — | — | — | ↕ | C = Dn |
| BNOT | — | — | — | — | — | |
| BOR | — | — | — | — | ↕ | C = C' + Dn |
| BSET | — | — | — | — | — | |
| BSR | — | — | — | — | — | |
| BST | — | — | — | — | — | |
| BTST | — | — | ↕ | — | — | Z = $\overline{Dn}$ |
| BXOR | — | — | — | — | ↕ | C = C' · $\overline{Dn}$ + $\overline{C'}$ · Dn |
| CLRMAC | — | — | — | — | — | |

**Table A-7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| CMP | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| DAA | * | ↕ | ↕ | * | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | C: decimal arithmetic carry |
| DAS | * | ↕ | ↕ | * | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | C: decimal arithmetic borrow |
| DEC | — | ↕ | ↕ | ↕ | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = Dm · $\overline{Rm}$ |
| DIVXS | — | ↕ | ↕ | — | — | N = Sm · $\overline{Dm}$ + $\overline{Sm}$ · Dm |
| | | | | | | Z = $\overline{Sm}$ · $\overline{Sm–1}$ · ····· · $\overline{S0}$ |
| DIVXU | — | ↕ | ↕ | — | — | N = Sm |
| | | | | | | Z = $\overline{Sm}$ · $\overline{Sm–1}$ · ····· · $\overline{S0}$ |
| EEPMOV | — | — | — | — | — | |
| EXTS | — | ↕ | ↕ | 0 | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| EXTU | — | 0 | ↕ | 0 | — | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| INC | — | ↕ | ↕ | ↕ | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Dm}$ · Rm |
| JMP | — | — | — | — | — | |
| JSR | — | — | — | — | — | |
| LDC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| | | | | | | No flags change when the operand is EXR. |
| LDM | — | — | — | — | — | |
| LDMAC | — | — | — | — | — | |
| MAC | — | — | — | — | — | |

**Table A-7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| MOV | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| MOVFPE | | | | | | Can not be used in the H8S/2655 Series. |
| MOVTPE | | | | | | |
| MULXS | — | ↕ | ↕ | — | — | N = R2m<br>Z = $\overline{R2m} \cdot \overline{R2m-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| MULXU | — | — | — | — | — | |
| NEG | ↕ | ↕ | ↕ | ↕ | ↕ | H = Dm–4 + Rm–4<br>N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>V = Dm · Rm<br>C = Dm + Rm |
| NOP | — | — | — | — | — | |
| NOT | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| OR | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| ORC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result.<br>No flags change when the operand is EXR. |
| POP | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| PUSH | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| ROTL | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| ROTR | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = D0 (1-bit shift) or C = D1 (2-bit shift) |

**Table A-7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| ROTXL | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| ROTXR | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| RTE | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| RTS | — | — | — | — | — | |
| SHAL | — | ↕ | ↕ | ↕ | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>V = $\overline{Dm \cdot Dm-1 + \overline{Dm} \cdot \overline{Dm-1}}$ (1-bit shift)<br>V = $\overline{Dm \cdot Dm-1 \cdot Dm-2 \cdot \overline{Dm} \cdot \overline{Dm-1} \cdot \overline{Dm-2}}$ (2-bit shift)<br>C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| SHAR | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| SHLL | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| SHLR | — | 0 | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| SLEEP | — | — | — | — | — | |
| STC | — | — | — | — | — | |
| STM | — | — | — | — | — | |
| STMAC | — | ↕ | ↕ | ↕ | — | N = 1 if MAC instruction resulted in negative value in MAC register<br>Z = 1 if MAC instruction resulted in zero value in MAC register<br>V = 1 if MAC instruction resulted in overflow |

**Table A-7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| SUB | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| SUBS | — | — | — | — | — | |
| SUBX | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = Z' · $\overline{Rm}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| TAS | — | ↕ | ↕ | 0 | — | N = Dm |
| | | | | | | Z = $\overline{Dm}$ · $\overline{Dm–1}$ · ····· · $\overline{D0}$ |
| TRAPA | — | — | — | — | — | |
| XOR | — | ↕ | ↕ | 0 | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| XORC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| | | | | | | No flags change when the operand is EXR. |

# Appendix B Internal I/O Register

## B.1 Addresses

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'F800 to H'FBFF | MRA | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz | DTC | 16/32* bit |
| | SAR | | | | | | | | | | |
| | MRB | CHNE | DISEL | — | — | — | — | — | — | | |
| | DAR | | | | | | | | | | |
| | CRA | | | | | | | | | | |
| | CRB | | | | | | | | | | |
| H'FE80 | TCR3 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU3 | 16 bit |
| H'FE81 | TMDR3 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | | |
| H'FE82 | TIOR3H | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FE83 | TIOR3L | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | | |
| H'FE84 | TIER3 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | | |
| H'FE85 | TSR3 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | | |
| H'FE86 | TCNT3 | | | | | | | | | | |
| H'FE87 | | | | | | | | | | | |
| H'FE88 | TGR3A | | | | | | | | | | |
| H'FE89 | | | | | | | | | | | |
| H'FE8A | TGR3B | | | | | | | | | | |
| H'FE8B | | | | | | | | | | | |
| H'FE8C | TGR3C | | | | | | | | | | |
| H'FE8D | | | | | | | | | | | |
| H'FE8E | TGR3D | | | | | | | | | | |
| H'FE8F | | | | | | | | | | | |

Note: * Located in on-chip RAM. The bus width is 32 bits when the DTC accesses this area as register information, and 16 bits otherwise.

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FE90 | TCR4 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU4 | 16 bit |
| H'FE91 | TMDR4 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FE92 | TIOR4 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FE94 | TIER4 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FE95 | TSR4 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FE96 | TCNT4 | | | | | | | | | | |
| H'FE97 | | | | | | | | | | | |
| H'FE98 | TGR4A | | | | | | | | | | |
| H'FE99 | | | | | | | | | | | |
| H'FE9A | TGR4B | | | | | | | | | | |
| H'FE9B | | | | | | | | | | | |
| H'FEA0 | TCR5 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU5 | 16 bit |
| H'FEA1 | TMDR5 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FEA2 | TIOR5 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FEA4 | TIER5 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FEA5 | TSR5 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FEA6 | TCNT5 | | | | | | | | | | |
| H'FEA7 | | | | | | | | | | | |
| H'FEA8 | TGR5A | | | | | | | | | | |
| H'FEA9 | | | | | | | | | | | |
| H'FEAA | TGR5B | | | | | | | | | | |
| H'FEAB | | | | | | | | | | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FEB0 | P1DDR | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR | Port | 8 bit |
| H'FEB1 | P2DDR | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR | | |
| H'FEB2 | P3DDR | — | — | P35DDR | P34DDR | P33DDR | P32DDR | P31DDR | P30DDR | | |
| H'FEB4 | P5DDR | — | — | — | — | P53DDR | P52DDR | P51DDR | P50DDR | | |
| H'FEB5 | P6DDR | P67DDR | P66DDR | P65DDR | P64DDR | P63DDR | P62DDR | P61DDR | P60DDR | | |
| H'FEB9 | PADDR | PA7DDR | PA6DDR | PA5DDR | PA4DDR | PA3DDR | PA2DDR | PA1DDR | PA0DDR | | |
| H'FEBA | PBDDR | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR | | |
| H'FEBB | PCDDR | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR | | |
| H'FEBC | PDDDR | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR | | |
| H'FEBD | PEDDR | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR | | |
| H'FEBE | PFDDR | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR | | |
| H'FEBF | PGDDR | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR | | |
| H'FEC0 | ICRA | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 | Interrupt controller | 8 bit |
| H'FEC1 | ICRB | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 | | |
| H'FEC2 | ICRC | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 | | |
| H'FEC4 | IPRA | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC5 | IPRB | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC6 | IPRC | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC7 | IPRD | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC8 | IPRE | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC9 | IPRF | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECA | IPRG | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECB | IPRH | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECC | IPRI | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECD | IPRJ | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECE | IPRK | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FED0 | ABWCR | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 | Bus controller | 8 bit |
| H'FED1 | ASTCR | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 | | |
| H'FED2 | WCRH | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 | | |
| H'FED3 | WCRL | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 | | |
| H'FED4 | BCRH | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | RMTS2 | RMTS1 | RMST0 | | |
| H'FED5 | BCRL | BLE | BREQOE | EAE | — | — | ASS | WDBE | WAITE | | |
| H'FED6 | MCR | TPC | BE | RCDM | CW2 | MXC1 | MXC0 | RLW1 | RLW0 | | |
| H'FED7 | DRAMCR | RFSHE | RCW | RMODE | CMF | CMIE | CKS2 | CKS1 | CKS0 | | |
| H'FED8 | RTCNT | | | | | | | | | | |
| H'FED9 | RTCOR | | | | | | | | | | |
| H'FEE0 | MAR0AH | — | — | — | — | — | — | — | — | DMAC | 16 bit |
| H'FEE1 | | | | | | | | | | | |
| H'FEE2 | MAR0AL | | | | | | | | | | |
| H'FEE3 | | | | | | | | | | | |
| H'FEE4 | IOAR0A | | | | | | | | | | |
| H'FEE5 | | | | | | | | | | | |
| H'FEE6 | ETCR0A | | | | | | | | | | |
| H'FEE7 | | | | | | | | | | | |
| H'FEE8 | MAR0BH | — | — | — | — | — | — | — | — | | |
| H'FEE9 | | | | | | | | | | | |
| H'FEEA | MAR0BL | | | | | | | | | | |
| H'FEEB | | | | | | | | | | | |
| H'FEEC | IOAR0B | | | | | | | | | | |
| H'FEED | | | | | | | | | | | |
| H'FEEE | ETCR0B | | | | | | | | | | |
| H'FEEF | | | | | | | | | | | |
| H'FEF0 | MAR1AH | — | — | — | — | — | — | — | — | | |
| H'FEF1 | | | | | | | | | | | |
| H'FEF2 | MAR1AL | | | | | | | | | | |
| H'FEF3 | | | | | | | | | | | |
| H'FEF4 | IOAR1A | | | | | | | | | | |
| H'FEF5 | | | | | | | | | | | |
| H'FEF6 | ETCR1A | | | | | | | | | | |
| H'FEF7 | | | | | | | | | | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FEF8 | MAR1BH | — | — | — | — | — | — | — | — | DMAC | 16 bit |
| H'FEF9 | | | | | | | | | | | |
| H'FEFA | MAR1BL | | | | | | | | | | |
| H'FEFB | | | | | | | | | | | |
| H'FEFC | IOAR1B | | | | | | | | | | |
| H'FEFD | | | | | | | | | | | |
| H'FEFE | ETCR1B | | | | | | | | | | |
| H'FEFF | | | | | | | | | | | |
| H'FF00 | DMAWER | — | — | — | — | WE1B | WE1A | WE0B | WE0A | | 8 bit |
| H'FF01 | DMATCR | — | — | TEE1 | TEE0 | — | — | — | — | | |
| H'FF02 | DMACR0A | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | Short address mode | 16 bit |
| | | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — | Full address mode | |
| H'FF03 | DMACR0B | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | Short address mode | |
| | | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 | Full address mode | |
| H'FF04 | DMACR1A | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | Short address mode | |
| | | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — | Full address mode | |
| H'FF05 | DMACR1B | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | Short address mode | |
| | | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 | Full address mode | |
| H'FF06 | DMABCRH | FAE1 | FAE0 | SAE1 | SAE0 | DTA1B | DTA1A | DTA0B | DTA0A | Short address mode | |
| | | FAE1 | FAE0 | — | — | DTA1 | — | DTA0 | — | Full address mode | |
| H'FF07 | DMABCRL | DTE1B | DTE1A | DTE0B | DTE0A | DTIE1B | DTIE1A | DTIE0B | DTIE0A | Short address mode | |
| | | DTME1 | DTE1 | DTME0 | DTE0 | DTIE1B | DTIE1A | DTIE0B | DTIE0A | Full address mode | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF2C | ISCRH | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA | Interrupt controller | 8 bit |
| H'FF2D | ISCRL | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA | | |
| H'FF2E | IER | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E | | |
| H'FF2F | ISR | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F | | |
| H'FF30 to H'FF35 | DTCER | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 | DTC | 8 bit |
| H'FF37 | DTVECR | SWDTE | DTVEC6 | DTVEC5 | DTVEC4 | DTVEC3 | DTVEC2 | DTVEC1 | DTVEC0 | | |
| H'FF38 | SBYCR | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — | | 8 bit |
| H'FF39 | SYSCR | MACS | — | INTM1 | INTM0 | NMIEG | — | — | RAME | | |
| H'FF3A | SCKCR | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 | | |
| H'FF3B | MDCR | — | — | — | — | — | MDS2 | MDS1 | MDS0 | | |
| H'FF3C | MSTPCRH | MSTP15 | MSTP14 | MSTP13 | MSTP12 | MSTP11 | MSTP10 | MSTP9 | MSTP8 | | |
| H'FF3D | MSTPCRL | MSTP7 | MSTP6 | MSTP5 | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 | | |
| H'FF46 | PCR | G3CMS1 | G3CMS0 | G2CMS1 | G2CMS0 | G1CMS1 | G1CMS0 | G0CMS1 | G0CMS0 | PPG | 8 bit |
| H'FF47 | PMR | G3INV | G2INV | G1INV | G0INV | G3NOV | G2NOV | G1NOV | G0NOV | | |
| H'FF48 | NDERH | NDER15 | NDER14 | NDER13 | NDER12 | NDER11 | NDER10 | NDER9 | NDER8 | | |
| H'FF49 | NDERL | NDER7 | NDER6 | NDER5 | NDER4 | NDER3 | NDER2 | NDER1 | NDER0 | | |
| H'FF4A | PODRH | POD15 | POD14 | POD13 | POD12 | POD11 | POD10 | POD9 | POD8 | | |
| H'FF4B | PODRL | POD7 | POD6 | POD5 | POD4 | POD3 | POD2 | POD1 | POD0 | | |
| H'FF4C* | NDRH | NDR15 | NDR14 | NDR13 | NDR12 | NDR11 | NDR10 | NDR9 | NDR8 | | |
| H'FF4D* | NDRL | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 | | |
| H'FF4E* | NDRH | — | — | — | — | NDR11 | NDR10 | NDR9 | NDR8 | | |
| H'FF4F* | NDRL | — | — | — | — | NDR3 | NDR2 | NDR1 | NDR0 | | |

Note: * If the pulse output group 2 and pulse output group 3 output triggers are the same according to the PCR setting, the NDRH address will be H'FF4C, and if different, the address of NDRH for group 2 will be H'FF4E, and that for group 3 will be H'FF4C. Similarly, if the pulse output group 0 and pulse output group 1 output triggers are the same according to the PCR setting, the NDRL address will be H'FF4D, and if different, the address of NDRL for group 0 will be H'FF4F, and that for group 1 will be H'FF4D.

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF50 | PORT1 | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | Port | 8 bit |
| H'FF51 | PORT2 | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 | | |
| H'FF52 | PORT3 | — | — | P35 | P34 | P33 | P32 | P31 | P30 | | |
| H'FF53 | PORT4 | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 | | |
| H'FF54 | PORT5 | — | — | — | — | P53 | P52 | P51 | P50 | | |
| H'FF55 | PORT6 | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 | | |
| H'FF59 | PORTA | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 | | |
| H'FF5A | PORTB | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 | | |
| H'FF5B | PORTC | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | | |
| H'FF5C | PORTD | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | | |
| H'FF5D | PORTE | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 | | |
| H'FF5E | PORTF | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 | | |
| H'FF5F | PORTG | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 | | |
| H'FF60 | P1DR | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR | | |
| H'FF61 | P2DR | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR | | |
| H'FF62 | P3DR | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR | | |
| H'FF64 | P5DR | — | — | — | — | P53DR | P52DR | P51DR | P50DR | | |
| H'FF65 | P6DR | P67DR | P66DR | P65DR | P64DR | P63DR | P62DR | P61DR | P60DR | | |
| H'FF69 | PADR | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR | | |
| H'FF6A | PBDR | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR | | |
| H'FF6B | PCDR | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR | | |
| H'FF6C | PDDR | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR | | |
| H'FF6D | PEDR | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR | | |
| H'FF6E | PFDR | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR | | |
| H'FF6F | PGDR | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR | | |
| H'FF70 | PAPCR | PA7PCR | PA6PCR | PA5PCR | PA4PCR | PA3PCR | PA2PCR | PA1PCR | PA0PCR | | |
| H'FF71 | PBPCR | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR | | |
| H'FF72 | PCPCR | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR | | |
| H'FF73 | PDPCR | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR | | |
| H'FF74 | PEPCR | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR | | |
| H'FF76 | P3ODR | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR | | |
| H'FF77 | PAODR | PA7ODR | PA6ODR | PA5ODR | PA4ODR | PA3ODR | PA2ODR | PA1ODR | PA0ODR | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF78 | SMR0 | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI0, Smart card interface 0 | 8 bit |
| H'FF79 | BRR0 | | | | | | | | | | |
| H'FF7A | SCR0 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | | |
| H'FF7B | TDR0 | | | | | | | | | | |
| H'FF7C | SSR0 | TDRE | RDRF | ORER | FER/ ERS | PER | TEND | MPB | MPBT | | |
| H'FF7D | RDR0 | | | | | | | | | | |
| H'FF7E | SCMR0 | — | — | — | — | SDIR | SINV | — | SMIF | | |
| H'FF80 | SMR1 | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI1, Smart card interface 1 | 8 bit |
| H'FF81 | BRR1 | | | | | | | | | | |
| H'FF82 | SCR1 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | | |
| H'FF83 | TDR1 | | | | | | | | | | |
| H'FF84 | SSR1 | TDRE | RDRF | ORER | FER/ ERS | PER | TEND | MPB | MPBT | | |
| H'FF85 | RDR1 | | | | | | | | | | |
| H'FF86 | SCMR1 | — | — | — | — | SDIR | SINV | — | SMIF | | |
| H'FF88 | SMR2 | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI2, Smart card interface 2 | 8 bit |
| H'FF89 | BRR2 | | | | | | | | | | |
| H'FF8A | SCR2 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | | |
| H'FF8B | TDR2 | | | | | | | | | | |
| H'FF8C | SSR2 | TDRE | RDRF | ORER | FER/ ERS | PER | TEND | MPB | MPBT | | |
| H'FF8D | RDR2 | | | | | | | | | | |
| H'FF8E | SCMR2 | — | — | — | — | SDIR | SINV | — | SMIF | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF90 | ADDRAH | — | — | — | — | — | — | AD9 | AD8 | A/D converter | 16 bit |
| H'FF91 | ADDRAL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF92 | ADDRBH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF93 | ADDRBL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF94 | ADDRCH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF95 | ADDRCL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF96 | ADDRDH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF97 | ADDRDL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF98 | ADDREH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF99 | ADDREL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF9A | ADDRFH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF9B | ADDRFL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF9C | ADDRGH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF9D | ADDRGL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FF9E | ADDRHH | — | — | — | — | — | — | AD9 | AD8 | | |
| H'FF9F | ADDRHL | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | | |
| H'FFA0 | ADCSR | ADF | ADIE | ADST | CKS | GRP | CH2 | CH1 | CH0 | | |
| H'FFA1 | ADCR | — | PWR | TRGS1 | TRGS0 | SCAN | DSMP | BUFE1 | BUFE0 | | |
| H'FFA4 | DADR0 | | | | | | | | | D/A converter | 8 bit |
| H'FFA5 | DADR1 | | | | | | | | | | |
| H'FFA6 | DACR | DAOE1 | DAOE0 | DAE | — | — | — | — | — | | |
| H'FFB0 | TCR0 | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 | 8-bit timer channel 0, 1 | 16 bit |
| H'FFB1 | TCR1 | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 | | |
| H'FFB2 | TCSR0 | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 | | |
| H'FFB3 | TCSR1 | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 | | |
| H'FFB4 | TCORA0 | | | | | | | | | | |
| H'FFB5 | TCORA1 | | | | | | | | | | |
| H'FFB6 | TCORB0 | | | | | | | | | | |
| H'FFB7 | TCORB1 | | | | | | | | | | |
| H'FFB8 | TCNT0 | | | | | | | | | | |
| H'FFB9 | TCNT1 | | | | | | | | | | |
| H'FFBC (read) | TCSR | OVF | WT/$\overline{IT}$ | TME | — | — | CKS2 | CKS1 | CKS0 | WDT | 16 bit |
| H'FFBD (read) | TCNT | | | | | | | | | | |
| H'FFBF (read) | RSTCSR | WOVF | RSTE | RSTS | — | — | — | — | — | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FFC0 | TSTR | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 | TPU | 16 bit |
| H'FFC1 | TSYR | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 | | |
| H'FFD0 | TCR0 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU0 | 16 bit |
| H'FFD1 | TMDR0 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | | |
| H'FFD2 | TIOR0H | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFD3 | TIOR0L | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | | |
| H'FFD4 | TIER0 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | | |
| H'FFD5 | TSR0 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | | |
| H'FFD6 | TCNT0 | | | | | | | | | | |
| H'FFD7 | | | | | | | | | | | |
| H'FFD8 | TGR0A | | | | | | | | | | |
| H'FFD9 | | | | | | | | | | | |
| H'FFDA | TGR0B | | | | | | | | | | |
| H'FFDB | | | | | | | | | | | |
| H'FFDC | TGR0C | | | | | | | | | | |
| H'FFDD | | | | | | | | | | | |
| H'FFDE | TGR0D | | | | | | | | | | |
| H'FFDF | | | | | | | | | | | |
| H'FFE0 | TCR1 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU1 | 16 bit |
| H'FFE1 | TMDR1 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FFE2 | TIOR1 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFE4 | TIER1 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FFE5 | TSR1 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FFE6 | TCNT1 | | | | | | | | | | |
| H'FFE7 | | | | | | | | | | | |
| H'FFE8 | TGR1A | | | | | | | | | | |
| H'FFE9 | | | | | | | | | | | |
| H'FFEA | TGR1B | | | | | | | | | | |
| H'FFEB | | | | | | | | | | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FFF0 | TCR2 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU2 | 16 bit |
| H'FFF1 | TMDR2 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FFF2 | TIOR2 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFF4 | TIER2 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FFF5 | TSR2 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FFF6 | TCNT2 | | | | | | | | | | |
| H'FFF7 | | | | | | | | | | | |
| H'FFF8 | TGR2A | | | | | | | | | | |
| H'FFF9 | | | | | | | | | | | |
| H'FFFA | TGR2B | | | | | | | | | | |
| H'FFFB | | | | | | | | | | | |

## B.2 Functions

**MRA—DTC Mode Register A** **H'F800—H'FBFF** **DTC**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz |
| Initial value | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| Read/Write | — | — | — | — | — | — | — | — |

DTC Data Transfer Size

| | |
|---|---|
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

DTC Transfer Mode Select

| | |
|---|---|
| 0 | Destination side is repeat area or block area |
| 1 | Source side is repeat area or block area |

DTC Mode

| | | |
|---|---|---|
| 0 | 0 | Normal mode |
| | 1 | Repeat mode |
| 1 | 0 | Block transfer mode |
| | 1 | — |

Destination Address Mode

| | | |
|---|---|---|
| 0 | — | DAR is fixed |
| 1 | 0 | DAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | DAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1) |

Source Address Mode

| | | |
|---|---|---|
| 0 | — | SAR is fixed |
| 1 | 0 | SAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | SAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1) |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHNE | DISEL | — | — | — | — | — | — |
| Initial value | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| Read/Write | — | — | — | — | — | — | — | — |

DTC Interrupt Select

| | |
|---|---|
| 0 | After a data transfer ends, the CPU interrupt is disabled unless the transfer counter is 0 |
| 1 | After a data transfer ends, the CPU interrupt is enabled |

DTC Chain Transfer Enable

| | |
|---|---|
| 0 | End of DTC data transfer |
| 1 | DTC chain transfer |



| Bit | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | - - - | | | | | |
| Initial value | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| Read/Write | — | — | — | — | — | - - - | — | — | — | — | — |

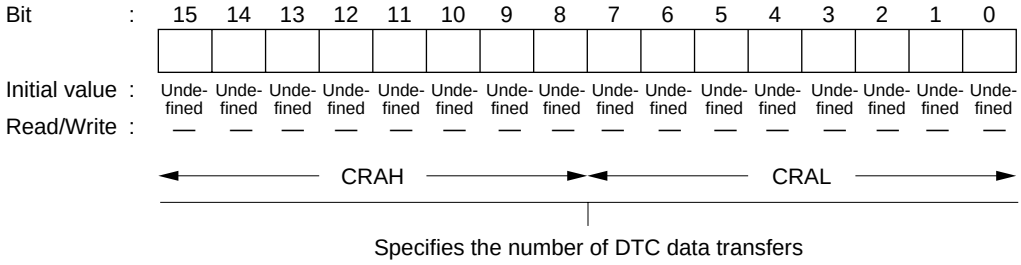
Specifies transfer data source address



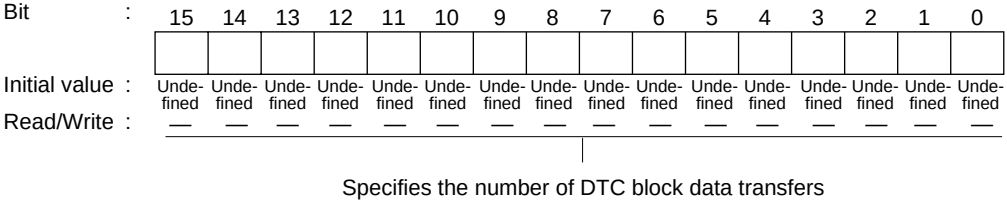
| Bit | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | - - - | | | | | |
| Initial value | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| Read/Write | — | — | — | — | — | - - - | — | — | — | — | — |

Specifies transfer data destination address

**CRA—DTC Transfer Count Register A** **H'F800—H'FBFF** **DTC**




**CRB—DTC Transfer Count Register B** **H'F800—H'FBFF** **DTC**



| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | Internal clock: counts on ø/1024 |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | Internal clock: counts on ø/4096 |

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Counter Clear

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRA compare match/input capture |
| | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation *1 |
| 1 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRC compare match/input capture *2 |
| | 1 | 0 | TCNT cleared by TGRD compare match/input capture *2 |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation *1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

Buffer Operation A

| | |
|---|---|
| 0 | TGRA operates normally |
| 1 | TGRA and TGRC used together for buffer operation |

Buffer Operation B

| | |
|---|---|
| 0 | TGRB operates normally |
| 1 | TGRB and TGRD used together for buffer operation |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Initial value : | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR3A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3A is input capture register | Capture input source is TIOCA3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

TGR3B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3B is input capture register | Capture input source is TIOCB3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000, and ø/1 is used as the TCNT4 count clock, this setting will be invalid and input capture will not occur.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TRG3C I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3C is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3C is input capture register | Capture input source is TIOCC3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

Note: When the BFA bit in TMDR3 is set to 1 and TGR3C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

TGR3D I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3D is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3D is input capture register *2 | Capture input source is TIOCD3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down*1 |

* : Don't care

Notes: 1 When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

2 When the BFB bit in TMDR3 is set to 1 and TGR3D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Note: When GRC or GRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

TGR Interrupt Enable C

| | |
|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled |

TGR Interrupt Enable D

| | |
|---|---|
| 0 | Interrupt requests (TGID) by TGFD bit disabled |
| 1 | Interrupt requests (TGID) by TGFD bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting condition]<br>• When TCNT=TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Input Capture/Output Compare Flag C

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

Input Capture/Output Compare Flag D

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting condition]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Note: * Can only be written with 0 for flag clearing.

**TCNT3—Timer Counter 3** **H'FE86** **TPU3**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up-counter

**TGR3A—Timer General Register 3A** **H'FE88** **TPU3**
**TGR3B—Timer General Register 3B** **H'FE8A** **TPU3**
**TGR3C—Timer General Register 3C** **H'FE8C** **TPU3**
**TGR3D—Timer General Register 3D** **H'FE8E** **TPU3**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKC pin input |
| | 1 | 0 | Internal clock: counts on ø/1024 |
| | | 1 | Counts on TCNT5 overflow/underflow |

Note: This setting is ignored when channel 4 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Note: This setting is ignored when channel 4 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR4A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR4A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR4A is input capture register | Capture input source is TIOCA4 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR3A compare match/ input capture | Input capture at generation of TGR3A compare match/input capture |

* : Don't care

TGR4B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR4B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR4B is input capture register | Capture input source is TIOCB4 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR3C compare match/ input capture | Input capture at generation of TGR3C compare match/input capture |

* : Don't care

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT4—Timer Counter 4** **H'FE96** **TPU4**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR4A—Timer General Register 4A** **H'FE98** **TPU4**

**TGR4B—Timer General Register 4B** **H'FE9A** **TPU4**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKC pin input |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | External clock: counts on TCLKD pin input |

Note: This setting is ignored when channel 5 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Note: This setting is ignored when channel 5 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR5A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR5A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR5A is input capture register | Capture input source is TIOCA5 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

TGR5B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR5B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR5B is input capture register | Capture input source is TIOCB5 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT5—Timer Counter 5** **H'FEA6** **TPU5**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR5A—Timer General Register 5A** **H'FEA8** **TPU5**
**TGR5B—Timer General Register 5B** **H'FEAA** **TPU5**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**P1DDR—Port 1 Data Direction Register** **H'FEB0** **Port 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port 1 pins

**P2DDR—Port 2 Data Direction Register** **H'FEB1** **Port 2**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port 2 pins

**P3DDR—Port 3 Data Direction Register** **H'FEB2** **Port 3**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35DDR | P34DDR | P33DDR | P32DDR | P31DDR | P30DDR |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | W | W | W | W | W | W |

Specify input or output for individual port 3 pins

**P5DDR—Port 5 Data Direction Register** **H'FEB4** **Port 5**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | P53DDR | P52DDR | P51DDR | P50DDR |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | W | W | W | W |

Specify input or output for individual port 5 pins

**P6DDR—Port 6 Data Direction Register H'FEB5 Port 6**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P67DDR | P66DDR | P65DDR | P64DDR | P63DDR | P62DDR | P61DDR | P60DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port 6 pins

**PADDR—Port A Data Direction Register H'FEB9 Port A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PA7DDR | PA6DDR | PA5DDR | PA4DDR | PA3DDR | PA2DDR | PA1DDR | PA0DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port A pins

**PBDDR—Port B Data Direction Register H'FEBA Port B**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port B pins

**PCDDR—Port C Data Direction Register** **H'FEBB** **Port C**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port C pins

**PDDDR—Port D Data Direction Register** **H'FEBC** **Port D**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port D pins

**PEDDR—Port E Data Direction Register** **H'FEBD** **Port E**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port E pins

**PFDDR—Port F Data Direction Register H'FEBE Port F**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR |
| Modes 1, 2, 4 to 6 | | | | | | | | | |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |
| Modes 3, 7 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port F pins

**PGDDR—Port G Data Direction Register H'FEBF Port G**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR |
| Modes 1, 4, 5 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | W | W | W | W | W |
| Modes 2, 3, 6, 7 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | W | W | W | W | W |

Specify input or output for individual port G pins

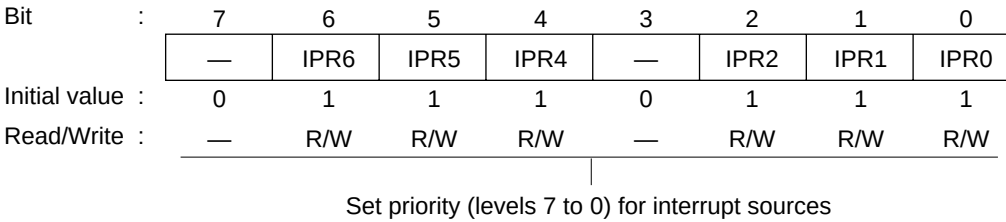
| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sets the interrupt control level for interrupts

Correspondence between Interrupt Sources and ICR Settings

| Register | Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ICRA | $IRQ_0$ | $IRQ_1$ | $IRQ_2$ $IRQ_3$ | $IRQ_4$ $IRQ_5$ | $IRQ_6$ $IRQ_7$ | DTC | Watchdog timer | Refresh timer |
| ICRB | — | A/D converter | TPU channel 0 | TPU channel 1 | TPU channel 2 | TPU channel 3 | TPU channel 4 | TPU channel 5 |
| ICRC | 8-bit timer channel 0 | 8-bit timer channel 1 | DMAC | SCI channel 0 | SCI channel 1 | SCI channel 2 | — | — |

| | | | |
|---|---|---|---|
| **IPRA** | **— Interrupt Priority Register A** | **H'FEC4** | **Interrupt Controller** |
| **IPRB** | **— Interrupt Priority Register B** | **H'FEC5** | **Interrupt Controller** |
| **IPRC** | **— Interrupt Priority Register C** | **H'FEC6** | **Interrupt Controller** |
| **IPRD** | **— Interrupt Priority Register D** | **H'FEC7** | **Interrupt Controller** |
| **IPRE** | **— Interrupt Priority Register E** | **H'FEC8** | **Interrupt Controller** |
| **IPRF** | **— Interrupt Priority Register F** | **H'FEC9** | **Interrupt Controller** |
| **IPRG** | **— Interrupt Priority Register G** | **H'FECA** | **Interrupt Controller** |
| **IPRH** | **— Interrupt Priority Register H** | **H'FECB** | **Interrupt Controller** |
| **IPRI** | **— Interrupt Priority Register I** | **H'FECC** | **Interrupt Controller** |
| **IPRJ** | **— Interrupt Priority Register J** | **H'FECD** | **Interrupt Controller** |
| **IPRK** | **— Interrupt Priority Register K** | **H'FECE** | **Interrupt Controller** |




Correspondence between Interrupt Sources and IPR Settings

| Register | Bits | |
|---|---|---|
| | 6 to 4 | 2 to 0 |
| IPRA | $IRQ_0$ | $IRQ_1$ |
| IPRB | $IRQ_2$<br>$IRQ_3$ | $IRQ_4$<br>$IRQ_5$ |
| IPRC | $IRQ_6$<br>$IRQ_7$ | DTC |
| IPRD | WDT | Refresh timer |
| IPRE | — | A/D converter |
| IPRF | TPU channel 0 | TPU channel 1 |
| IPRG | TPU channel 2 | TPU channel 3 |
| IPRH | TPU channel 4 | TPU channel 5 |
| IPRI | 8-bit timer channel 0 | 8-bit timer channel 1 |
| IPRJ | DMAC | SCI channel 0 |
| IPRK | SCI channel 1 | SCI channel 2 |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 |
| Modes 1, 2, 5, 6 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Modes 3, 4, 7 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 7 to 0 Bus Width Control

| | |
|---|---|
| 0 | Area n is designated for 16-bit access |
| 1 | Area n is designated for 8-bit access |

(n = 7 to 0)

**ASTCR—Access State Control Register** **H'FED1** **Bus Controller**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 7 to 0 Access State Control

| | |
|---|---|
| 0 | Area n is designated for 2-state access<br>Wait state insertion in area n external space is disabled |
| 1 | Area n is designated for 3-state access<br>Wait state insertion in area n external space is enabled |

(n = 7 to 0)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 |
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 4 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 5 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 6 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 7 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 |
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 0 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 1 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 2 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 3 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | RMTS2 | RMTS1 | RMTS0 |
| Initial value | : | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

RAM Type Select

| RMTS2 | RMTS1 | RMTS0 | Area 5 | Area 4 | Area 3 | Area 2 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | Normal space | | | |
| | | 1 | Normal space | | | DRAM space |
| | 1 | 0 | Normal space | | DRAM space | |
| | | 1 | DRAM space | | | |
| 1 | 0 | 0 | Normal space | | | |
| | | 1 | Normal space | | | PSRAM space |
| | 1 | 0 | Normal space | | PSRAM space | |
| | | 1 | PSRAM space | | | |

Burst Cycle Select 0

| | |
|---|---|
| 0 | Max. 4 words in burst access |
| 1 | Max. 8 words in burst access |

Burst Cycle Select 1

| | |
|---|---|
| 0 | Burst cycle comprises 1 state |
| 1 | Burst cycle comprises 2 states |

Area 0 Burst ROM Enable

| | |
|---|---|
| 0 | Area 0 is basic bus interface |
| 1 | Area 0 is burst ROM interface |

Idle Cycle Insert 0

| | |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read and external write cycles |
| 1 | Idle cycle inserted in case of successive external read and external write cycles |

Idle Cycle Insert 1

| | |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read cycles in different areas |
| 1 | Idle cycle inserted in case of successive external read cycles in different areas |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | BRLE | BREQOE | EAE | LCASS | DDS | ASS | WDBE | WAITE |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

WAIT Pin Enable

| | |
|---|---|
| 0 | Wait input by $\overline{\text{WAIT}}$ pin disabled |
| 1 | Wait input by $\overline{\text{WAIT}}$ pin enabled |

Write Data Buffer Enable

| | |
|---|---|
| 0 | Write data buffer function not used |
| 1 | Write data buffer function used |

Area Partition Unit Select

| | |
|---|---|
| 0 | Area partition unit is 128 kbytes (1 Mbit) |
| 1 | Area partition unit is 2 Mbytes (16 Mbits) |

DACK Timing Select

| | |
|---|---|
| 0 | When DMAC single address transfer is performed in DRAM/PSRAM space, full access is always executed $\overline{\text{DACK}}$ signal goes low from Tr or T1 cycle |
| 1 | Burst access is possible when DMAC single address transfer is performed in DRAM/PSRAM space $\overline{\text{DACK}}$ signal goes low from Tc1 or T2 cycle |

LCAS Pin Select

| | |
|---|---|
| 0 | $\overline{\text{LCAS}}$ pin used for 2-CAS type DRAM interface $\overline{\text{LCAS}}$ signal ($\overline{\text{BREQO}}$ output and $\overline{\text{WAIT}}$ input cannot be used when $\overline{\text{LCAS}}$ signal is used) |
| 1 | $\overline{\text{LWR}}$ pin used for 2-CAS type DRAM interface $\overline{\text{LCAS}}$ signal (RAS down mode cannot be used) |

External Addresses H'010000 to H'01FFFF Enable

| | |
|---|---|
| 0 | On-chip ROM (H8S/2655) or reserved area* (H8S/2653) |
| 1 | External addresses (in external expansion mode) or reserved area (in single-chip mode) |

Notes: * Do not access a reserved area.

BREQO Pin Enable

| | |
|---|---|
| 0 | $\overline{\text{BREQO}}$ output disabled |
| 1 | $\overline{\text{BREQO}}$ output enabled |

Bus Release Enable

| | |
|---|---|
| 0 | External bus release is disabled |
| 1 | External bus release is enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TPC | BE | RCDM | CW2 | MXC1 | MXC0 | RLW1 | RLW0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Refresh Cycle Wait Control

| | | |
|---|---|---|
| 0 | 0 | No wait state inserted |
| | 1 | 1 wait state inserted |
| 1 | 0 | 2 wait states inserted |
| | 1 | 3 wait states inserted |

Multiplex Shift Count

| | | |
|---|---|---|
| 0 | 0 | 8-bit shift |
| | 1 | 9-bit shift |
| 1 | 0 | 10-bit shift |
| | 1 | — |

2-CAS Method/2-WE Method Select

| | |
|---|---|
| 0 | 2-CAS method selected: $\overline{\text{CASH}}$, $\overline{\text{CASL}}$, $\overline{\text{WE}}$ signals enabled. |
| 1 | 2-WE method selected: $\overline{\text{CAS}}$, $\overline{\text{UWE}}$, $\overline{\text{LWE}}$ signals enabled. |

RAS/CS Down Mode

| | |
|---|---|
| 0 | DRAM interface: RAS up mode selected |
| 1 | DRAM interface: RAS down mode selected |

Burst Access Enable

| | |
|---|---|
| 0 | Burst disabled (always full access) |
| 1 | • For DRAM space access<br>Access in fast page mode<br>• For PSRAM space access<br>Access in static column mode |

TP Cycle Control

| | |
|---|---|
| 0 | 1-state precharge cycle is inserted |
| 1 | 2-state precharge cycle is inserted |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RFSHE | RCW | RMODE | CMF | CMIE | CKS2 | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Refresh Counter Clock Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Count operation disabled |
| | | 1 | Count uses ø/2 |
| | 1 | 0 | Count uses ø/8 |
| | | 1 | Count uses ø/32 |
| 1 | 0 | 0 | Count uses ø/128 |
| | | 1 | Count uses ø/512 |
| | 1 | 0 | Count uses ø/2048 |
| | | 1 | Count uses ø/4096 |

Compare Match Interrupt Enable

| | |
|---|---|
| 0 | Interrupt request (CMI) by CMF flag disabled |
| 1 | Interrupt request (CMI) by CMF flag enabled |

Compare Match Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading the CMF flag when CMF = 1, then writing 0 to the CMF flag |
| 1 | [Setting condition]<br>Set when RTCNT = RTCOR |

Refresh Mode

| | |
|---|---|
| 0 | DRAM interface: CAS-before-RAS refreshing used |
| | PSRAM interface: Auto-refreshing used |
| 1 | Self-refreshing used |

RAS-CAS Wait

| | |
|---|---|
| 0 | Wait state insertion in CAS-before-RAS refreshing disabled<br>$\overline{RAS}$ falls in $T_{Rr}$ cycle |
| 1 | One wait state inserted in CAS-before-RAS refreshing<br>$\overline{RAS}$ falls in $T_{Rc1}$ cycle |

Refresh Control

| | |
|---|---|
| 0 | Refresh control is not performed |
| 1 | Refresh control is performed |

**RTCNT—Refresh Timer Counter** **H'FED8** **Bus Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Internal clock count value

**RTCOR—Refresh Time Constant Register** **H'FED9** **Bus Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sets the period for compare match operations with RTCNT

**MAR0AH—Memory Address Register 0AH** **H'FEE0** **DMAC**
**MAR0AL—Memory Address Register 0AL** **H'FEE2** **DMAC**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR0AH | | | | | | | | | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| Read/Write | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR0AL | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Specifies transfer source address

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IOAR0A | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Not used

**ETCR0A—Transfer Count Register 0A** **H'FEE6** **DMAC**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR0A | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | |
|---|---|---|
| Sequential mode<br>Idle mode<br>Normal mode | Transfer counter | |
| Repeat mode | Transfer number storage register | Transfer counter |
| Block transfer mode | Block size storage register | Block size counter |

* : Undefined

**MAR0BH—Memory Address Register 0BH** **H'FEE8** **DMAC**
**MAR0BL—Memory Address Register 0BL** **H'FEEA** **DMAC**

| Bit | : | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR0BH | : | — | — | — | — | — | — | — | — | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| Read/Write | : | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR0BL | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Specifies transfer destination address

**IOAR0B—I/O Address Register 0B** **H'FEEC** **DMAC**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IOAR0B | : | | | | | | | | | | | | | | | | |
| Initial value | : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Not used

**ETCR0B—Transfer Count Register 0B** **H'FEEE** **DMAC**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR0B | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sequential mode and idle mode: Transfer counter

Repeat mode: Transfer number storage register / Transfer counter

Block transfer mode: Block transfer counter

* : Undefined

Note: Not used in normal mode.

**MAR1AH—Memory Address Register 1AH** **H'FEF0** **DMAC**
**MAR1AL—Memory Address Register 1AL** **H'FEF2** **DMAC**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR1AH | — | — | — | — | — | — | — | — | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| Read/Write | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR1AL | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Specifies transfer source address

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IOAR1A | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Not used



| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR1A | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sequential mode
Idle mode
Normal mode

Transfer counter

Repeat mode

Transfer number storage register | Transfer counter

Block transfer mode

Block size storage register | Block size counter

* : Undefined

**MAR1BH — Memory Address Register 1BH** **H'FEF8** **DMAC**
**MAR1BL — Memory Address Register 1BL** **H'FEFA** **DMAC**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR1BH | — | — | — | — | — | — | — | — | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | * | * | * | * | * | * | * |
| Read/Write | — | — | — | — | — | — | — | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR1BL | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Specifies transfer destination address

**IOAR1B—I/O Address Register 1B** **H'FEFC** **DMAC**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IOAR1B | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

* : Undefined

In short address mode: Specifies transfer source/transfer destination address
In full address mode: Not used

**ETCR1B—Transfer Count Register 1B** **H'FEFE** **DMAC**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETCR1B | | | | | | | | | | | | | | | | |
| Initial value | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sequential mode and idle mode: Transfer counter

Repeat mode: Transfer number storage register / Transfer counter

Block transfer mode: Block transfer counter

* : Undefined

Note: Not used in normal mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMAWER | — | — | — | — | WE1B | WE1A | WE0B | WE0A |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Write Enable 0A

| | |
|---|---|
| 0 | Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are disabled |
| 1 | Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are enabled |

Write Enable 0B

| | |
|---|---|
| 0 | Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are disabled |
| 1 | Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are enabled |

Write Enable 1A

| | |
|---|---|
| 0 | Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are disabled |
| 1 | Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are enabled |

Write Enable 1B

| | |
|---|---|
| 0 | Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are disabled |
| 1 | Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMATCR | — | — | TEE1 | TEE0 | — | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | — | — | — | — |

Transfer End Enable 0

| | |
|---|---|
| 0 | $\overline{\text{TEND}}_0$ pin output disabled |
| 1 | $\overline{\text{TEND}}_0$ pin output enabled |

Transfer End Enable 1

| | |
|---|---|
| 0 | $\overline{\text{TEND}}_1$ pin output disabled |
| 1 | $\overline{\text{TEND}}_1$ pin output enabled |

Full address mode

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| DMACRA | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Block Direction/Block Enable

| | | |
|---|---|---|
| 0 | 0 | Transfer in normal mode |
| | 1 | Transfer in block transfer mode, destination side is block area |
| 1 | 0 | Transfer in normal mode |
| | 1 | Transfer in block transfer mode, source side is block area |

Source Address Increment/Decrement

| | | |
|---|---|---|
| 0 | 0 | MARA is fixed |
| | 1 | MARA is incremented after a data transfer |
| 1 | 0 | MARA is fixed |
| | 1 | MARA is decremented after a data transfer |

Data Transfer Size

| | |
|---|---|
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

Full address mode (cont)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMACRB | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Data Transfer Factor

| DTF 3 | DTF 2 | DTF 1 | DTF 0 | Block Transfer Mode | Normal Mode |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | — | — |
| | | | 1 | Activated by A/D converter conversion end interrupt | — |
| | | 1 | 0 | Activated by $\overline{\text{DREQ}}$ pin falling edge input | Activated by $\overline{\text{DREQ}}$ pin falling edge input |
| | | | 1 | Activated by $\overline{\text{DREQ}}$ pin low-level input | Activated by $\overline{\text{DREQ}}$ pin low-level input |
| | 1 | 0 | 0 | Activated by SCI channel 0 transmission complete interrupt | — |
| | | | 1 | Activated by SCI channel 0 reception complete interrupt | — |
| | | 1 | 0 | Activated by SCI channel 1 transmission complete interrupt | Auto-request (cycle steal) |
| | | | 1 | Activated by SCI channel 1 reception complete interrupt | Auto-request (burst) |
| 1 | 0 | 0 | 0 | Activated by TPU channel 0 compare match/input capture A interrupt | — |
| | | | 1 | Activated by TPU channel 1 compare match/input capture A interrupt | — |
| | | 1 | 0 | Activated by TPU channel 2 compare match/input capture A interrupt | — |
| | | | 1 | Activated by TPU channel 3 compare match/input capture A interrupt | — |
| | 1 | 0 | 0 | Activated by TPU channel 4 compare match/input capture A interrupt | — |
| | | | 1 | Activated by TPU channel 5 compare match/input capture A interrupt | — |
| | | 1 | * | — | — |

* : Don't care

Destination Address Increment/Decrement

| | | |
|---|---|---|
| 0 | 0 | MARB is fixed |
| | 1 | MARB is incremented after a data transfer |
| 1 | 0 | MARB is fixed |
| | 1 | MARB is decremented after a data transfer |

Short address mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMACR | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Data Transfer Direction

| | |
|---|---|
| 0 | Dual address mode: Transfer with MAR as source address and IOAR as destination address<br>Single address mode: Transfer with MAR as source address and $\overline{\text{DACK}}$ pin as write strobe |
| 1 | Dual address mode: Transfer with IOAR as source address and MAR as destination address<br>Single address mode: Transfer with $\overline{\text{DACK}}$ pin as read strobe and MAR as destination address |

Repeat Enable

| | |
|---|---|
| 0 | Transfer in sequential mode |
| 1 | Transfer in repeat mode or idle mode |

Data Transfer Increment/Decrement

| | |
|---|---|
| 0 | MAR is incremented after a data transfer |
| 1 | MAR is decremented after a data transfer |

Data Transfer Size

| | |
|---|---|
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

Data Transfer Factor

| | | | | Channel A | Channel B |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | — | |
| | | | 1 | Activated by A/D converter conversion end interrupt | |
| | | 1 | 0 | — | Activated by $\overline{\text{DREQ}}$ pin falling edge input |
| | | | 1 | — | Activated by $\overline{\text{DREQ}}$ pin low-level input |
| | 1 | 0 | 0 | Activated by SCI channel 0 transmission complete interrupt | |
| | | | 1 | Activated by SCI channel 0 reception complete interrupt | |
| | | 1 | 0 | Activated by SCI channel 1 transmission complete interrupt | |
| | | | 1 | Activated by SCI channel 1 reception complete interrupt | |
| 1 | 0 | 0 | 0 | Activated by TPU channel 0 compare match/input capture A interrupt | |
| | | | 1 | Activated by TPU channel 1 compare match/input capture A interrupt | |
| | | 1 | 0 | Activated by TPU channel 2 compare match/input capture A interrupt | |
| | | | 1 | Activated by TPU channel 3 compare match/input capture A interrupt | |
| | 1 | 0 | 0 | Activated by TPU channel 4 compare match/input capture A interrupt | |
| | | | 1 | Activated by TPU channel 5 compare match/input capture A interrupt | |
| | | 1 | 0 | — | |
| | | | 1 | — | |

* : Don't care

Full address mode

| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| DMABCRH : | FAE1 | FAE0 | — | — | DTA1 | — | DTA0 | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel 0 Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 1 Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 0 Full Address Enable

| | |
|---|---|
| 0 | Short address mode |
| 1 | Full address mode |

Channel 1 Full Address Enable

| | |
|---|---|
| 0 | Short address mode |
| 1 | Full address mode |

(Continued on next page)

Full address mode (cont)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMABCRL | DTME1 | DTE1 | DTME0 | DTE0 | DTIE1B | DTIE1A | DTIE0B | DTIE0A |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel 0 Data Transfer Interrupt Enable A

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 0 Data Transfer Interrupt Enable B

| | |
|---|---|
| 0 | Transfer suspended interrupt disabled |
| 1 | Transfer suspended interrupt enabled |

Channel 0 Data Transfer Interrupt Enable A

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 1 Data Transfer Interrupt Enable B

| | |
|---|---|
| 0 | Transfer suspended interrupt disabled |
| 1 | Transfer suspended interrupt enabled |

Channel 0 Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

Channel 0 Data Transfer Master Enable

| | |
|---|---|
| 0 | Data transfer disabled. In normal mode, cleared to 0 by an NMI interrupt |
| 1 | Data transfer enabled |

Channel 1 Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

Channel 0 Data Transfer Master Enable

| | |
|---|---|
| 0 | Data transfer disabled. In normal mode, cleared to 0 by an NMI interrupt |
| 1 | Data transfer enabled |

(Continued on next page)

Short address mode

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRH | : | FAE1 | FAE0 | SAE1 | SAE0 | DTA1B | DTA1A | DTA0B | DTA0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel 0A Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 0B Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 1A Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 1B Data Transfer Acknowledge

| | |
|---|---|
| 0 | Clearing of selected internal interrupt source at time of DMA transfer is disabled |
| 1 | Clearing of selected internal interrupt source at time of DMA transfer is enabled |

Channel 0B Single Address Enable

| | |
|---|---|
| 0 | Transfer in dual address mode |
| 1 | Transfer in single address mode |

Channel 1B Single Address Enable

| | |
|---|---|
| 0 | Transfer in dual address mode |
| 1 | Transfer in single address mode |

Channel 0 Full Address Enable

| | |
|---|---|
| 0 | Short address mode |
| 1 | Full address mode |

Channel 1 Full Address Enable

| | |
|---|---|
| 0 | Short address mode |
| 1 | Full address mode |

(Continued on next page)

Short address mode (cont)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| DMABCRL | : | DTE1B | DTE1A | DTE0B | DTE0A | DTIE1B | DTIE1A | DTIE0B | DTIE0A |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel 0A Data Transfer Interrupt Enable

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 0B Data Transfer Interrupt Enable

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 1A Data Transfer Interrupt Enable

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 1B Data Transfer Interrupt Enable

| | |
|---|---|
| 0 | Transfer end interrupt disabled |
| 1 | Transfer end interrupt enabled |

Channel 0A Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

Channel 0B Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

Channel 1A Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

Channel 1B Data Transfer Enable

| | |
|---|---|
| 0 | Data transfer disabled |
| 1 | Data transfer enabled |

ISCRH

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

$IRQ_7$ to $IRQ_4$ Sense Control

ISCRL

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

$IRQ_3$ to $IRQ_0$ Sense Control

| $IRQ_n$SCB | $IRQ_n$SCA | Interrupt Request Generation |
|---|---|---|
| 0 | 0 | $\overline{IRQ}_n$ input low level |
| | 1 | Falling edge of $\overline{IRQ}_n$ input |
| 1 | 0 | Rising edge of $\overline{IRQ}_n$ input |
| | 1 | Both falling and rising edges of $\overline{IRQ}_n$ input |

(n = 7 to 0)

**IER—IRQ Enable Register** **H'FF2E** **Interrupt Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

$IRQ_n$ Enable

| | |
|---|---|
| 0 | $IRQ_n$ interrupt disabled |
| 1 | $IRQ_n$ interrupt enabled |

(n = 7 to 0)

**ISR—IRQ Status Register** **H'FF2F** **Interrupt Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Indicate the status of $IRQ_7$ to $IRQ_0$ interrupt requests

Note: * Can only be written with 0 for flag clearing.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DTC Activation Enable

| | |
|---|---|
| 0 | DTC activation by this interrupt is disabled<br>[Clearing conditions]<br>• When the DISEL bit is 1 and data transfer has ended<br>•When the specified number of transfers have ended |
| 1 | DTC activation by this interrupt is enabled<br>[Holding condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended |

### Correspondence between Interrupt Sources and DTCER

| | Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Register** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DTCERA | IRQ0 | IRQ1 | IRQ2 | IRQ3 | IRQ4 | IRQ5 | IRQ6 | IRQ7 |
| DTCERB | — | ADI | TGI0A | TGI0B | TGI0C | TGI0D | TGI1A | TGI1B |
| DTCERC | TGI2A | TGI2B | TGI3A | TGI3B | TGI3C | TGI3D | TGI4A | TGI4B |
| DTCERD | — | — | TGI5A | TGI5B | CMI0A | CMI0B | CMI1A | CMI1B |
| DTCERE | DMTEND0A | DMTEND0B | DMTEND1A | DMTEND1B | RXI0 | TXI0 | RXI1 | TXI1 |
| DTCERF | RXI2 | TXI2 | — | — | — | — | — | — |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SWDTE | DTVEC6 | DTVEC5 | DTVEC4 | DTVEC3 | DTVEC2 | DTVEC1 | DTVEC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Sets vector number for DTC software activation

DTC Software Activation Enable

| | |
|---|---|
| 0 | DTC software activation is disabled<br>[Clearing condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended |
| 1 | DTC software activation is enabled<br>[Holding conditions]<br>• When the DISEL bit is 1 and data transfer has ended<br>• When the specified number of transfers have ended<br>• During data transfer due to software activation |

Note: * A value of 1 can always be written to the SWDTE bit, but 0 can only be written after 1 is read.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | — | — | — |

Output Port Enable

| | |
|---|---|
| 0 | In software standby mode, address bus and bus control signals are high-impedance |
| 1 | In software standby mode, address bus and bus control signals retain output state |

Standby Timer Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Standby time = 8192 states |
| | | 1 | Standby time = 16384 states |
| | 1 | 0 | Standby time = 32768 states |
| | | 1 | Standby time = 65536 states |
| 1 | 0 | 0 | Standby time = 131072 states |
| | | 1 | Standby time = 262144 states |
| | 1 | 0 | Reserved |
| | | 1 | Standby time = 16 states |

Software Standby

| | |
|---|---|
| 0 | Transition to sleep mode after execution of SLEEP instruction |
| 1 | Transition to software standby mode after execution of SLEEP instruction |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MACS | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Read/Write | R/W | — | R/W | R/W | R/W | — | — | R/W |

RAM Enable

| | |
|---|---|
| 0 | On-chip RAM disabled |
| 1 | On-chip RAM enabled |

NMI Input Edge Select

| | |
|---|---|
| 0 | Falling edge |
| 1 | Rising edge |

Interrupt Control Mode Selection

| | | |
|---|---|---|
| 0 | 0 | Interrupt control mode 0 |
| | 1 | Interrupt control mode 1 |
| 1 | 0 | Interrupt control mode 2 |
| | 1 | Interrupt control mode 3 |

MAC Saturation

| | |
|---|---|
| 0 | Non-saturating calculation for MAC instruction |
| 1 | Saturating calculation for MAC instruction |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | — | — | — | R/W | R/W | R/W |

Bus Master Clock Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Bus master is in high-speed mode |
| | | 1 | Medium-speed clock is ø/2 |
| | 1 | 0 | Medium-speed clock is ø/4 |
| | | 1 | Medium-speed clock is ø/8 |
| 1 | 0 | 0 | Medium-speed clock is ø/16 |
| | | 1 | Medium-speed clock is ø/32 |
| | 1 | — | — |

ø Clock Output Control

| PSTOP | Normal Operation | Sleep Mode | Software Standby Mode | Hardware Standby Mode |
|---|---|---|---|---|
| 0 | ø output | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | MDS2 | MDS1 | MDS0 |
| Initial value | 1 | 0 | 0 | 0 | 0 | —* | —* | —* |
| Read/Write | — | — | — | — | — | R | R | R |

Current mode pin operating mode

Note: * Determined by pins $MD_2$ to $MD_0$

**MSTPCRH — Module Stop Control Register H H'FF3C Power-Down State**
**MSTPCRL — Module Stop Control Register L H'FF3D Power-Down State**

| | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Specifies module stop mode

| | |
|---|---|
| 0 | Module stop mode cleared |
| 1 | Module stop mode set |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | G3CMS1 | G3CMS0 | G2CMS1 | G2CMS0 | G1CMS1 | G1CMS0 | G0CMS1 | G0CMS0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Output Trigger for Pulse Output Group 0

| | | |
|---|---|---|
| 0 | 0 | Compare match in TPU channel 0 |
| | 1 | Compare match in TPU channel 1 |
| 1 | 0 | Compare match in TPU channel 2 |
| | 1 | Compare match in TPU channel 3 |

Output Trigger for Pulse Output Group 1

| | | |
|---|---|---|
| 0 | 0 | Compare match in TPU channel 0 |
| | 1 | Compare match in TPU channel 1 |
| 1 | 0 | Compare match in TPU channel 2 |
| | 1 | Compare match in TPU channel 3 |

Output Trigger for Pulse Output Group 2

| | | |
|---|---|---|
| 0 | 0 | Compare match in TPU channel 0 |
| | 1 | Compare match in TPU channel 1 |
| 1 | 0 | Compare match in TPU channel 2 |
| | 1 | Compare match in TPU channel 3 |

Output Trigger for Pulse Output Group 3

| | | |
|---|---|---|
| 0 | 0 | Compare match in TPU channel 0 |
| | 1 | Compare match in TPU channel 1 |
| 1 | 0 | Compare match in TPU channel 2 |
| | 1 | Compare match in TPU channel 3 |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | G3INV | G2INV | G1INV | G0INV | G3NOV | G2NOV | G1NOV | G0NOV |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Pulse Output Group n Normal/Non-Overlap Operation Select

| | |
|---|---|
| 0 | Normal operation in pulse output group n (output values updated at compare match A in the selected TPU channel) |
| 1 | Non-overlapping operation in pulse output group n (independent 1 and 0 output at compare match A or B in the selected TPU channel) |

n=3 to 0

Pulse Output Group n Direct/Inverted Output

| | |
|---|---|
| 0 | Inverted output for pulse output group n (low-level output at pin for a 1 in PODRH) |
| 1 | Direct output for pulse output group n (high-level output at pin for a 1 in PODRH) |

n=3 to 0

NDERH

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDER15 | NDER14 | NDER13 | NDER12 | NDER11 | NDER10 | NDER9 | NDER8 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Pulse Output Enable/Disable

| | |
|---|---|
| 0 | Pulse outputs $PO_{15}$ to $PO_8$ are disabled |
| 1 | Pulse outputs $PO_{15}$ to $PO_8$ are enabled |

NDERL

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | NDER7 | NDER6 | NDER5 | NDER4 | NDER3 | NDER2 | NDER1 | NDER0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Pulse Output Enable/Disable

| | |
|---|---|
| 0 | Pulse outputs $PO_7$ to $PO_0$ are disabled |
| 1 | Pulse outputs $PO_7$ to $PO_0$ are enabled |

PODRH

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | POD15 | POD14 | POD13 | POD12 | POD11 | POD10 | POD9 | POD8 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Stores output data for use in pulse output

PODRL

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | POD7 | POD6 | POD5 | POD4 | POD3 | POD2 | POD1 | POD0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Stores output data for use in pulse output

Note: * A bit that has been set for pulse output by NDER is read-only.

(1) When pulse output group output triggers are the same

(a) Address: H'FF4C

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDR15 | NDR14 | NDR13 | NDR12 | NDR11 | NDR10 | NDR9 | NDR8 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores the next data for pulse output groups 3 and 2

(b) Address: H'FF4E

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | — | — | — | — | — | — | — | — |

(2) When pulse output group output triggers are different

(a) Address: H'FF4C

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDR15 | NDR14 | NDR13 | NDR12 | — | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | — | — | — | — |

Stores the next data for pulse output group 3

(b) Address: H'FF4E

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | NDR11 | NDR10 | NDR9 | NDR8 |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Stores the next data for pulse output group 2

(1) When pulse output group output triggers are the same

(a) Address: H'FF4D

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores the next data for pulse output groups 1 and 0

(b) Address: H'FF4F

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | — | — | — | — | — | — | — | — |

(2) When pulse output group output triggers are different

(a) Address: H'FF4D

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDR7 | NDR6 | NDR5 | NDR4 | — | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | — | — | — | — |

Stores the next data for pulse output group 1

(b) Address: H'FF4F

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Stores the next data for pulse output group 0

**PORT1—Port 1 Register** **H'FF50** **Port 1**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port 1 pins

Note: * Determined by the state of pins $P1_7$ to $P1_0$.

**PORT2—Port 2 Register** **H'FF51** **Port 2**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port 2 pins

Note: * Determined by the state of pins $P2_7$ to $P2_0$.

**PORT3—Port 3 Register** **H'FF52** **Port 3**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35 | P34 | P33 | P32 | P31 | P30 |
| Initial value : | 1 | 1 | —* | —* | —* | —* | —* | —* |
| Read/Write : | — | — | R | R | R | R | R | R |

State of port 3 pins

Note: * Determined by the state of pins $P3_5$ to $P3_0$.

**PORT4—Port 4 Register** **H'FF53** **Port 4**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port 4 pins

Note: * Determined by the state of pins $P4_7$ to $P4_0$.

**PORT5—Port 5 Register** **H'FF54** **Port 5**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | P53 | P52 | P51 | P50 |
| Initial value | 1 | 1 | 1 | 1 | —* | —* | —* | —* |
| Read/Write | — | — | — | — | R | R | R | R |

State of port 5 pins

Note: * Determined by the state of pins $P5_3$ to $P5_0$.

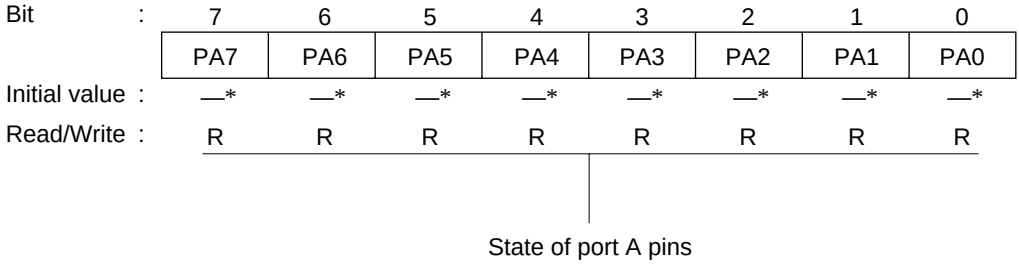
**PORT6—Port 6 Register** **H'FF55** **Port 6**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port 6 pins

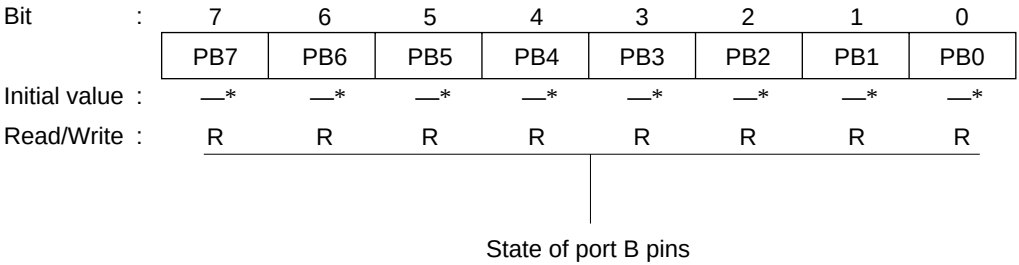
Note: * Determined by the state of pins $P6_7$ to $P6_0$.

**PORTA—Port A Register** **H'FF59** **Port A**



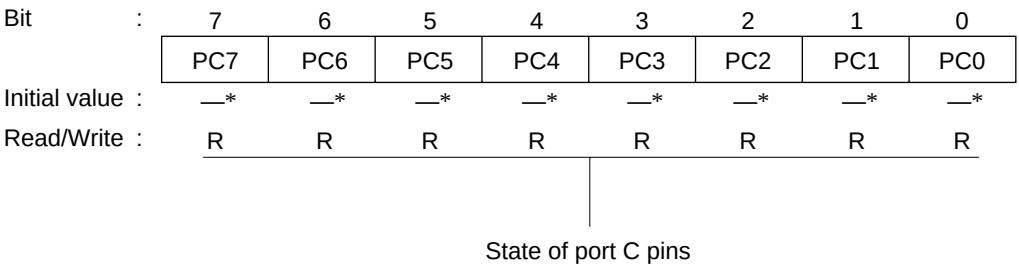

Note: * Determined by the state of pins $PA_7$ to $PA_0$.

**PORTB—Port B Register** **H'FF5A** **Port B**




Note: * Determined by the state of pins $PB_7$ to $PB_0$.

**PORTC—Port C Register** **H'FF5B** **Port C**




Note: * Determined by the state of pins $PC_7$ to $PC_0$.

**PORTD—Port D Register** **H'FF5C** **Port D**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port D pins

Note: * Determined by the state of pins $PD_7$ to $PD_0$.

**PORTE—Port E Register** **H'FF5D** **Port E**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port E pins

Note: * Determined by the state of pins $PE_7$ to $PE_0$.

**PORTF—Port F Register** **H'FF5E** **Port F**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port F pins

Note: * Determined by the state of pins $PF_7$ to $PF_0$.

**PORTG—Port G Register** **H'FF5F** **Port G**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 |
| Initial value | 1 | 1 | 1 | —* | —* | —* | —* | —* |
| Read/Write | — | — | — | R | R | R | R | R |

State of port G pins

Note: * Determined by the state of pins $PG_4$ to $PG_0$.

**P1DR—Port 1 Data Register** **H'FF60** **Port 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 1 pins ($P1_7$ to $P1_0$)

**P2DR—Port 2 Data Register** **H'FF61** **Port 2**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 2 pins ($P2_7$ to $P2_0$)

**P3DR—Port 3 Data Register** **H'FF62** **Port 3**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 3 pins ($P3_5$ to $P3_0$)

**P5DR—Port 5 Data Register** **H'FF64** **Port 5**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | P53DR | P52DR | P51DR | P50DR |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Stores output data for port 5 pins ($P5_3$ to $P5_0$)

**P6DR—Port 6 Data Register** **H'FF65** **Port 6**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P67DR | P66DR | P65DR | P64DR | P63DR | P62DR | P61DR | P60DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 6 pins ($P6_7$ to $P6_0$)

**PADR—Port A Data Register** **H'FF69** **Port A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port A pins ($PA_7$ to $PA_0$)

**PBDR—Port B Data Register** **H'FF6A** **Port B**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port B pins ($PB_7$ to $PB_0$)

**PCDR—Port C Data Register** **H'FF6B** **Port C**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port C pins ($PC_7$ to $PC_0$)

**PDDR—Port D Data Register** **H'FF6C** **Port D**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port D pins ($PD_7$ to $PD_0$)

**PEDR—Port E Data Register** **H'FF6D** **Port E**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port E pins ($PE_7$ to $PE_0$)

**PFDR—Port F Data Register** **H'FF6E** **Port F**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port F pins ($PF_7$ to $PF_0$)

**PGDR—Port G Data Register H'FF6F Port G**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR |
| Initial value | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | R/W | R/W | R/W | R/W | R/W |

Stores output data for port G pins ($PG_4$ to $PG_0$)

**PAPCR—Port A MOS Pull-Up Control Register H'FF70 Port A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PA7PCR | PA6PCR | PA5PCR | PA4PCR | PA3PCR | PA2PCR | PA1PCR | PA0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port A on a bit-by-bit basis

**PBPCR—Port B MOS Pull-Up Control Register H'FF71 Port B**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port B on a bit-by-bit basis

**PCPCR—Port C MOS Pull-Up Control Register H'FF72** **Port C**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port C on a bit-by-bit basis

**PDPCR—Port D MOS Pull-Up Control Register H'FF73** **Port D**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port D on a bit-by-bit basis

**PEPCR—Port E MOS Pull-Up Control Register H'FF74** **Port E**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port E on a bit-by-bit basis

**P3ODR—Port 3 Open Drain Control Register H'FF76 Port 3**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the PMOS on/off status for each port 3 pin ($P3_5$ to $P3_0$)

**PAODR—Port A Open Drain Control Register H'FF77 Port A**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PA7ODR | PA6ODR | PA5ODR | PA4ODR | PA3ODR | PA2ODR | PA1ODR | PA0ODR |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the PMOS on/off status for each port A pin ($PA_7$ to $PA_0$)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

Asynchronous Mode/Synchronous Mode Select

| | |
|---|---|
| 0 | Asynchronous mode |
| 1 | Synchronous mode |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

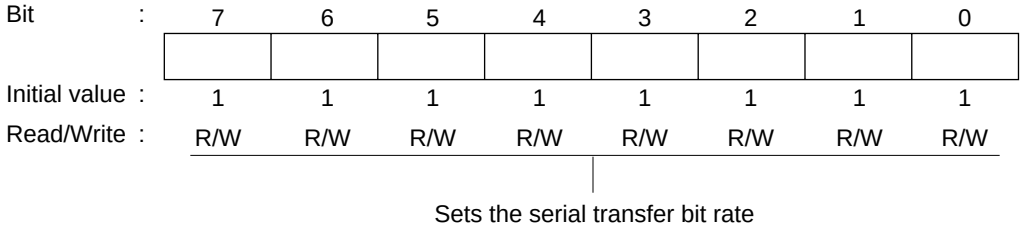
| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

GSM Mode

| | |
|---|---|
| 0 | Normal smart card interface mode operation<br>• TEND flag generated 12.5 etu after beginning of start bit<br>• Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generated 11.0 etu after beginning of start bit<br>• Fixed high/low-level control possible (set in SCR) in addition to clock output on/off control |

Note: etu (Elementary Time Unit): Interval for transfer of one bit




Note: See section 14.2.8, Bit Rate Register (BRR), for details.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| | | | |
|---|---|---|---|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*1 |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Outputs a clock of the same frequency as the bit rate.
2. Inputs a clock with a frequency 16 times the bit rate.

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| SMCR | SMR | SCR setting | | SCK pin function |
|---|---|---|---|---|
| SMIF | C/$\overline{A}$, GM | CKE1 | CKE0 | |
| 0 | See SCI specification | | | |
| 1 | 0 | 0 | 0 | Operates as port input pin |
| 1 | 0 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 0 | 0 | Fixed-low output as SCK output pin |
| 1 | 1 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 1 | 0 | Fixed-high output as SCK output pin |
| 1 | 1 | 1 | 1 | Clock output as SCK output pin |

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

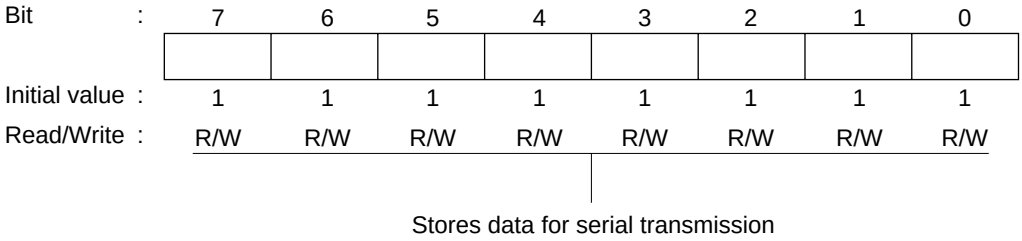
| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

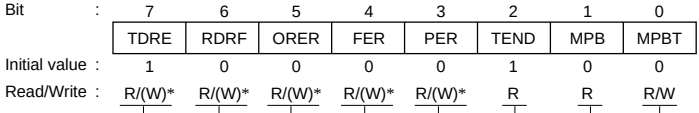
Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |


Bit :
7
6
5
4
3
2
1
0
Initial value :
1
1
1
1
1
1
1
1
Read/Write :
R/W
R/W
R/W
R/W
R/W
R/W
R/W
R/W
Stores data for serial transmission



| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Framing Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to FER after reading FER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0 |

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions]<br>• On reset, or in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When TDRE = 1 and ERS = 0, 2.5 etu after a 1-byte serial transmit character is sent (normal transmission) |

Note: etu: Elementary Time Unit (the time taken to transmit one bit)

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Error Signal Status

| | |
|---|---|
| 0 | [Clearing condition]<br>• On reset, or in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS = 1 |
| 1 | [Setting condition]<br>When the error signal is sampled at the low level |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its prior state.

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

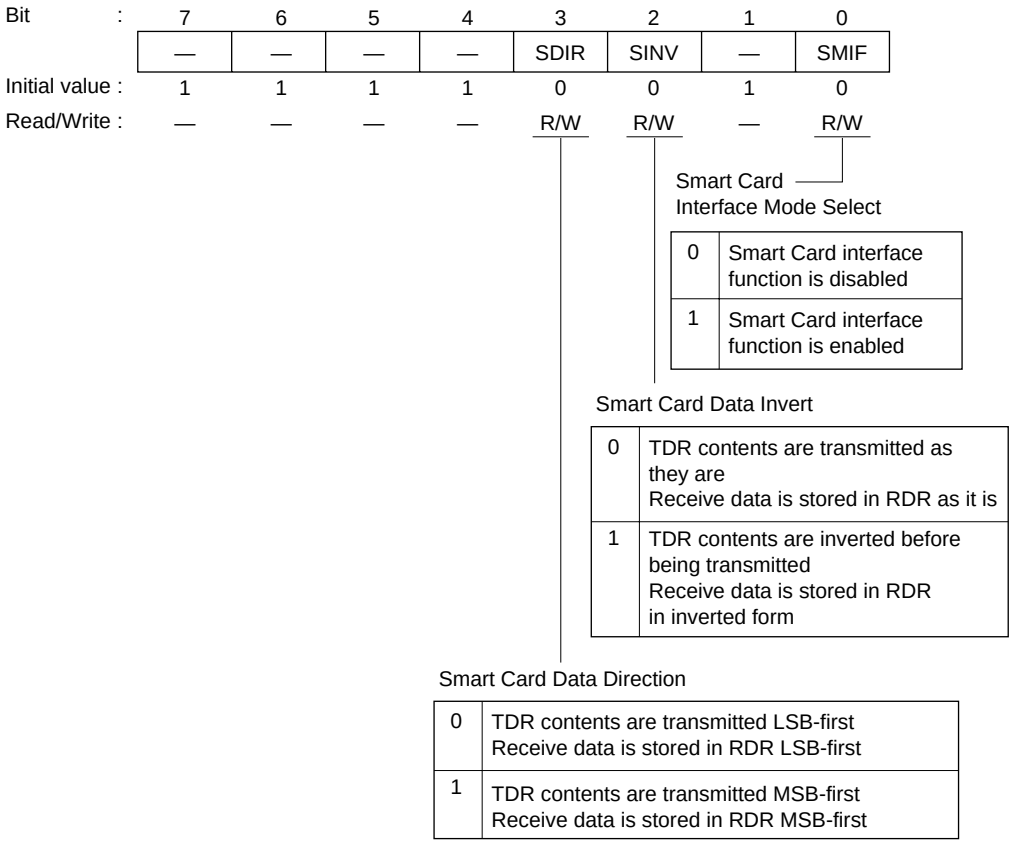
Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

**RDR0—Receive Data Register 0** **H'FF7D** **SCI0, Smart Card Interface 0**

**SCMR0—Smart Card Mode Register 0** **H'FF7E** **SCI0, Smart Card Interface 0**



| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

Asynchronous Mode/Synchronous Mode Select

| | |
|---|---|
| 0 | Asynchronous mode |
| 1 | Synchronous mode |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

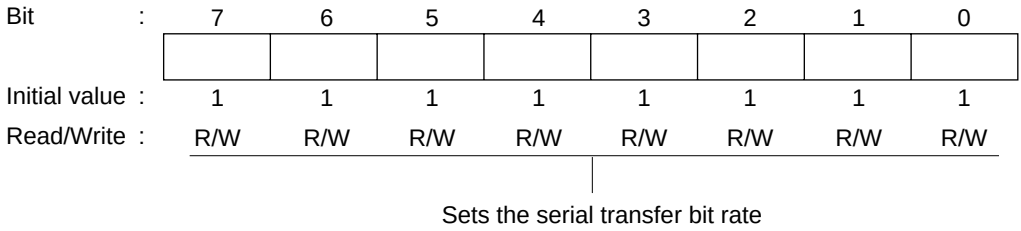
| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

GSM Mode

| | |
|---|---|
| 0 | Normal smart card interface mode operation<br>• TEND flag generated 12.5 etu after beginning of start bit<br>• Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generated 11.0 etu after beginning of start bit<br>• Fixed high/low-level control possible (set in SCR) in addition to clock output on/off control |

Note: etu (Elementary Time Unit): Interval for transfer of one bit




Note: See section 14.2.8, Bit Rate Register (BRR), for details.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| | | | |
|---|---|---|---|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*1 |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Outputs a clock of the same frequency as the bit rate.
2. Inputs a clock with a frequency 16 times the bit rate.

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| SMCR | SMR | SCR setting | | SCK pin function |
|---|---|---|---|---|
| SMIF | C/$\overline{A}$, GM | CKE1 | CKE0 | |
| 0 | See SCI specification | | | |
| 1 | 0 | 0 | 0 | Operates as port input pin |
| 1 | 0 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 0 | 0 | Fixed-low output as SCK output pin |
| 1 | 1 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 1 | 0 | Fixed-high output as SCK output pin |
| 1 | 1 | 1 | 1 | Clock output as SCK output pin |

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores data for serial transmission

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/E bit in SMR |

Framing Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to FER after reading FER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0 |

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting conditions]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions]<br>• On reset, or in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When TDRE = 1 and ERS = 0, 2.5 etu after a 1-byte serial transmit character is sent (normal transmission) |

Note: etu: Elementary Time Unit (the time taken to transmit one bit)

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Error Signal Status

| | |
|---|---|
| 0 | [Clearing condition]<br>• On reset, or in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS =1 |
| 1 | [Setting conditions]<br>When the error signal is sampled at the low level |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its prior state.

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

**RDR1—Receive Data Register 1** **H'FF85 SCI1, Smart Card Interface 1**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R | R | R | R | R | R | R | R |

Stores received serial data

**SCMR1—Smart Card Mode Register 1** **H'FF86 SCI1, Smart Card Interface 1**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | — | R/W |

Smart Card Interface Mode Select

| | |
|---|---|
| 0 | Smart Card interface function is disabled |
| 1 | Smart Card interface function is enabled |

Smart Card Data Invert

| | |
|---|---|
| 0 | TDR contents are transmitted as they are<br>Receive data is stored in RDR as it is |
| 1 | TDR contents are inverted before being transmitted<br>Receive data is stored in RDR in inverted form |

Smart Card Data Direction

| | |
|---|---|
| 0 | TDR contents are transmitted LSB-first<br>Receive data is stored in RDR LSB-first |
| 1 | TDR contents are transmitted MSB-first<br>Receive data is stored in RDR MSB-first |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

Asynchronous Mode/Synchronous Mode Select

| | |
|---|---|
| 0 | Asynchronous mode |
| 1 | Synchronous mode |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

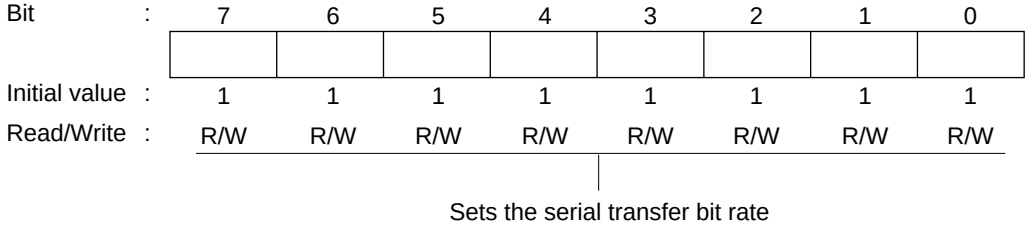
| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

GSM Mode

| | |
|---|---|
| 0 | Normal smart card interface mode operation<br>• TEND flag generated 12.5 etu after beginning of start bit<br>• Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generated 11.0 etu after beginning of start bit<br>• Fixed high/low-level control possible (set in SCR) in addition to clock output on/off control |

Note: etu (Elementary Time Unit): Interval for transfer of one bit




Note: See section 14.2.8, Bit Rate Register (BRR), for details.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| | | | |
|---|---|---|---|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*1 |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Outputs a clock of the same frequency as the bit rate.
2. Inputs a clock with a frequency 16 times the bit rate.

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| SMCR | SMR | SCR setting | | SCK pin function |
|---|---|---|---|---|
| SMIF | C/$\overline{A}$, GM | CKE1 | CKE0 | |
| 0 | See SCI specification | | | |
| 1 | 0 | 0 | 0 | Operates as port input pin |
| 1 | 0 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 0 | 0 | Fixed-low output as SCK output pin |
| 1 | 1 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 1 | 0 | Fixed-high output as SCK output pin |
| 1 | 1 | 1 | 1 | Clock output as SCK output pin |

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores data for serial transmission

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Framing Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0 |

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting conditions]<br>• On reset, or in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When TDRE = 1 and ERS = 0, 2.5 etu after a 1-byte serial transmit character is sent (normal transmission) |

Note: etu: Elementary Time Unit (the time taken to transmit one bit)

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Error Signal Status

| | |
|---|---|
| 0 | [Clearing condition]<br>• On reset, or in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS =1 |
| 1 | [Setting conditions]<br>When the error signal is sampled at the low level |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its prior state.

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting conditions]<br>When the next serial reception completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DMAC or DTC is activated by an RXI interrupt and read data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC or DTC is activated by a TXI interrupt and write data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | R | R | R | R | R | R | R |

Stores received serial data

## SCMR2—Smart Card Mode Register 2 H'FF8E SCI2, Smart Card Interface 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | — | R/W |

Smart Card Interface Mode Select

| | |
|---|---|
| 0 | Smart Card interface function is disabled |
| 1 | Smart Card interface function is enabled |

Smart Card Data Invert

| | |
|---|---|
| 0 | TDR contents are transmitted as they are<br>Receive data is stored in RDR as it is |
| 1 | TDR contents are inverted before being transmitted<br>Receive data is stored in RDR in inverted form |

Smart Card Data Direction

| | |
|---|---|
| 0 | TDR contents are transmitted LSB-first<br>Receive data is stored in RDR LSB-first |
| 1 | TDR contents are transmitted MSB-first<br>Receive data is stored in RDR MSB-first |

| | | | | |
|---|---|---|---|---|
| **ADDRAH** | **—** | **A/D Data Register AH** | **H'FF90** | **A/D Converter** |
| **ADDRAL** | **—** | **A/D Data Register AL** | **H'FF91** | **A/D Converter** |
| **ADDRBH** | **—** | **A/D Data Register BH** | **H'FF92** | **A/D Converter** |
| **ADDRBL** | **—** | **A/D Data Register BL** | **H'FF93** | **A/D Converter** |
| **ADDRCH** | **—** | **A/D Data Register CH** | **H'FF94** | **A/D Converter** |
| **ADDRCL** | **—** | **A/D Data Register CL** | **H'FF95** | **A/D Converter** |
| **ADDRDH** | **—** | **A/D Data Register DH** | **H'FF96** | **A/D Converter** |
| **ADDRDL** | **—** | **A/D Data Register DL** | **H'FF97** | **A/D Converter** |
| **ADDREH** | **—** | **A/D Data Register EH** | **H'FF98** | **A/D Converter** |
| **ADDREL** | **—** | **A/D Data Register EL** | **H'FF99** | **A/D Converter** |
| **ADDRFH** | **—** | **A/D Data Register FH** | **H'FF9A** | **A/D Converter** |
| **ADDRFL** | **—** | **A/D Data Register FL** | **H'FF9B** | **A/D Converter** |
| **ADDRGH** | **—** | **A/D Data Register GH** | **H'FF9C** | **A/D Converter** |
| **ADDRGL** | **—** | **A/D Data Register GL** | **H'FF9D** | **A/D Converter** |
| **ADDRHH** | **—** | **A/D Data Register HH** | **H'FF9E** | **A/D Converter** |
| **ADDRHL** | **—** | **A/D Data Register HL** | **H'FF9F** | **A/D Converter** |

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | — | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | — | — | R | R | R | R | R | R | R | R | R | R |

Stores the results of A/D conversion

| Analog Input Channel | A/D Data Register |
|---|---|
| $AN_0$ | ADDRA* |
| $AN_1$ | ADDRB* |
| $AN_2$ | ADDRC* |
| $AN_3$ | ADDRD* |
| $AN_4$ | ADDRE |
| $AN_5$ | ADDRF |
| $AN_6$ | ADDRG |
| $AN_7$ | ADDRH |

Note: * Except when buffer operation is used.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADF | ADIE | ADST | CKS | GRP | CH2 | CH1 | CH0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel Select

| CH2 | CH1 | CH0 | Select Mode (GRP = 0) | Group Mode (GRP = 1) |
|---|---|---|---|---|
| 0 | 0 | 0 | $AN_0$ | $AN_0$ |
| | | 1 | $AN_1$ | $AN_0$ to $AN_1$ |
| | 1 | 0 | $AN_2$ | $AN_0$ to $AN_2$ |
| | | 1 | $AN_3$ | $AN_0$ to $AN_3$ |
| 1 | 0 | 0 | $AN_4$ | $AN_0$ to $AN_4$ |
| | | 1 | $AN_5$ | $AN_0$ to $AN_5$ |
| | 1 | 0 | $AN_6$ | $AN_0$ to $AN_6$ |
| | | 1 | $AN_7$ | $AN_0$ to $AN_7$ |

Group Mode

| | |
|---|---|
| 0 | Select mode |
| 1 | Group mode |

Clock Select

| | |
|---|---|
| 0 | Conversion time = 20 states (A/D converter reference clock = ø) |
| 1 | Conversion time = 40 states (when ø/2 is selected) |

A/D Start

| | |
|---|---|
| 0 | A/D conversion stopped |
| 1 | • Single mode: A/D conversion is started. Cleared to 0 automatically when conversion on the specified channel ends<br>• Scan mode: A/D conversion is started. Conversion continues until ADST is cleared to 0 by software |

A/D Interrupt Enable

| | |
|---|---|
| 0 | A/D conversion end interrupt (ADI) request disabled |
| 1 | A/D conversion end interrupt (ADI) request enabled |

A/D End Flag

| | |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to the ADF flag after reading ADF = 1<br>• When the DMAC or DTC is activated by an ADI interrupt, and the relevant register is read |
| 1 | [Setting conditions]<br>• Single mode: When conversion ends for all specified channels, and A/D conversion ends*<br>• Scan mode: When one round of conversion has been performed on all specified channels* |

Note: * Can only be written with 0 for flag clearing.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | PWR | TRGS1 | TRGS0 | SCAN | DSMP | BUFE1 | BUFE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Buffer Enable

| | | |
|---|---|---|
| 0 | 0 | Normal operation |
| | 1 | ADDRA and ADDRB are used for buffer operation<br>(Conversion result → ADDRA → ADDRB)<br>ADDRB is buffer register |
| 1 | 0 | ADDRA and ADDRC, and ADDRB and ADDRD, are used for buffer operation<br>(Conversion result 1 → ADDRA → ADDRC; conversion result 2 → ADDRB → ADDRD)<br>ADDRC and ADDRD are buffer registers |
| | 1 | ADDRA to ADDRD are used for buffer operation<br>(Conversion result → ADDRA → ADDRB → ADDRC → ADDRD)<br>ADDRB to ADDRD are buffer registers |

Simultaneous Sampling

| | |
|---|---|
| 0 | Normal sampling operation |
| 1 | Simultaneous sampling operation |

Scan Mode

| | |
|---|---|
| 0 | Single mode |
| 1 | Scan mode |

Timer Trigger Select

| | | |
|---|---|---|
| 0 | 0 | A/D conversion start by software is enabled |
| | 1 | A/D conversion start by TPU conversion start trigger is enabled |
| 1 | 0 | A/D conversion start by 8-bit timer conversion start trigger is enabled |
| | 1 | A/D conversion start by external trigger pin ($\overline{\text{ADTRG}}$) is enabled |

Specifies conversion start mode

| | |
|---|---|
| 0 | Low-power conversion mode |
| 1 | High-speed start mode |

**DADR0—D/A Data Register 0** **H'FFA4** **D/A**
**DADR1—D/A Data Register 1** **H'FFA5** **D/A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DAOE1 | DAOE0 | DAE | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | — | — | — | — | — |

D/A Output Enable 0

| | |
|---|---|
| 0 | Analog output $DA_0$ is disabled |
| 1 | Channel 0 D/A conversion is enabled<br>Analog output $DA_0$ is enabled |

D/A Output Enable 1

| | |
|---|---|
| 0 | Analog output $DA_1$ is disabled |
| 1 | Channel 1 D/A conversion is enabled<br>Analog output $DA_1$ is enabled |

D/A Conversion Control

| | | | |
|---|---|---|---|
| 0 | 0 | * | Channel 0 and 1 D/A conversion disabled |
| | 1 | 0 | Channel 0 D/A conversion enabled |
| | | | Channel 1 D/A conversion disabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| 1 | 0 | 0 | Channel 0 D/A conversion disabled |
| | | | Channel 1 D/A conversion enabled |
| | | 1 | Channel 0 and 1 D/A conversion enabled |
| | 1 | * | Channel 0 and 1 D/A conversion enabled |

* : Don’t care

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Clock input disabled |
| | | 1 | Internal clock: counted at falling edge of ø/8 |
| | 1 | 0 | Internal clock: counted at falling edge of ø/64 |
| | | 1 | Internal clock: counted at falling edge of ø/8192 |
| 1 | 0 | 0 | For channel 0:<br>Count at TCNT1 overflow signal*<br>For channel 1:<br>Count at TCNT0 compare match A* |
| | | 1 | External clock: counted at rising edge |
| | 1 | 0 | External clock: counted at falling edge |
| | | 1 | External clock: counted at both rising and falling edges |

Note: * If the count input of channel 0 is the TCNT1 overflow signal and that of channel 1 is the TCNT0 compare match signal, no incrementing clock is generated. Do not use this setting.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | Clear is disabled |
| | 1 | Clear by compare match A |
| 1 | 0 | Clear by compare match B |
| | 1 | Clear by rising edge of external reset input |

Timer Overflow Interrupt Enable

| | |
|---|---|
| 0 | OVF interrupt requests (OVI) are disabled |
| 1 | OVF interrupt requests (OVI) are enabled |

Compare Match Interrupt Enable A

| | |
|---|---|
| 0 | CMFA interrupt requests (CMIA) are disabled |
| 1 | CMFA interrupt requests (CMIA) are enabled |

Compare Match Interrupt Enable B

| | |
|---|---|
| 0 | CMFB interrupt requests (CMIB) are disabled |
| 1 | CMFB interrupt requests (CMIB) are enabled |

TCSR0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/W | R/W | R/W | R/W | R/W |

TCSR1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 |
| Initial value | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | — | R/W | R/W | R/W | R/W |

Output Select

| | | |
|---|---|---|
| 0 | 0 | No change when compare match A occurs |
| | 1 | 0 is output when compare match A occurs |
| 1 | 0 | 1 is output when compare match A occurs |
| | 1 | Output is inverted when compare match A occurs (toggle output) |

Output Select

| | | |
|---|---|---|
| 0 | 0 | No change when compare match B occurs |
| | 1 | 0 is output when compare match B occurs |
| 1 | 0 | 1 is output when compare match B occurs |
| | 1 | Output is inverted when compare match B occurs (toggle output) |

A/D Trigger Enable (TCSR0 only)

| | |
|---|---|
| 0 | A/D converter start requests by compare match A are disabled |
| 1 | A/D converter start requests by compare match A are enabled |

Timer Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading OVF when OVF = 1, then writing 0 to OVF |
| 1 | [Setting condition]<br>Set when TCNT overflows (changes from H'FF to H'00) |

Compare Match Flag A

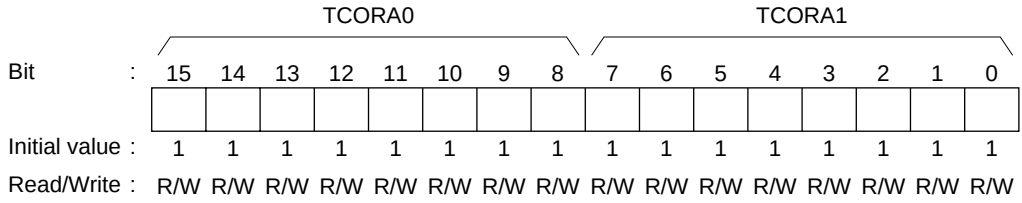
| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA<br>• When the DTC is activated by a CMIA interrupt, while DISEL bit of MRB in DTC is 0. |
| 1 | [Setting condition]<br>Set when TCNT matches TCORA |

Compare Match Flag B

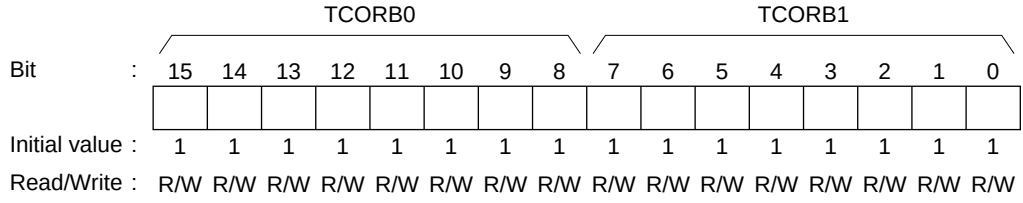
| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB<br>• When the DTC is activated by a CMIB interrupt, while DISEL bit of MRB in DTC is 0. |
| 1 | [Setting condition]<br>Set when TCNT matches TCORB |

Note: * Only 0 can be written to bits 7 to 5, to clear these flags.

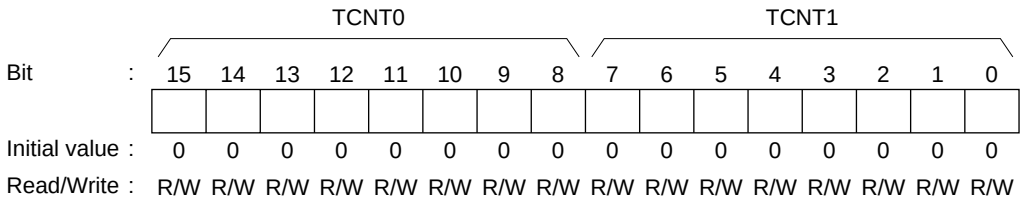
**TCORA0—Time Constant Register A0** **H'FFB4** **8-Bit Timer Channel 0**
**TCORA1—Time Constant Register A1** **H'FFB5** **8-Bit Timer Channel 1**




**TCORB0—Time Constant Register B0** **H'FFB6** **8-Bit Timer Channel 0**
**TCORB1—Time Constant Register B1** **H'FFB7** **8-Bit Timer Channel 1**




**TCNT0—Timer Counter 0** **H'FFB8** **8-Bit Timer Channel 0**
**TCNT1—Timer Counter 1** **H'FFB9** **8-Bit Timer Channel 1**



| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | WT/$\overline{IT}$ | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/W | R/W | — | — | R/W | R/W | R/W |

Clock Select

| CKS2 | CKS1 | CKS0 | Clock | Overflow period* (when ø = 20 MHz) |
|---|---|---|---|---|
| 0 | 0 | 0 | ø/2 (initial value) | 25.6µs |
| | | 1 | ø/64 | 819.2µs |
| | 1 | 0 | ø/128 | 1.6ms |
| | | 1 | ø/512 | 6.6ms |
| 1 | 0 | 0 | ø/2048 | 26.2ms |
| | | 1 | ø/8192 | 104.9ms |
| | 1 | 0 | ø/32768 | 419.4ms |
| | | 1 | ø/131072 | 1.68s |

Note: * The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.

Timer Enable

| | |
|---|---|
| 0 | TCNT is initialized to H'00 and halted |
| 1 | TCNT counts |

Timer Mode Select

| | |
|---|---|
| 0 | Interval timer mode: Sends the CPU an interval timer interrupt request (WOVI) when TCNT overflows |
| 1 | Watchdog timer mode: Generates the $\overline{WDTOVF}$ signal when TCNT overflows |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading TCSR when OVF = 1, then writing 0 to OVF |
| 1 | [Setting condition]<br>Set when TCNT overflows from H'FF to H'00 in interval timer mode |

The method for writing to TCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

Note: * Can only be written with 0 for flag clearing.

**TCNT—Timer Counter** **H'FFBC (W) H'FFBD (R)** **WDT**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The method for writing to TCNT is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

**RSTCSR—Reset Control/Status Register** **H'FFBE (W) H'FFBF (R)** **WDT**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WOVF | RSTE | RSTS | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/(W)* | R/W | R/W | — | — | — | — | — |

Reset Select

| | |
|---|---|
| 0 | Power-on reset |
| 1 | Manual reset |

Reset Enable

| | |
|---|---|
| 0 | Reset signal is not generated if TCNT overflows* |
| 1 | Reset signal is generated if TCNT overflows |

Note: * The modules H8S/2655 Series are not reset, but TCNT and TCSR in WDT are reset.

Watchdog Timer Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading TCSR when WOVF = 1, then writing 0 to WOVF |
| 1 | [Setting condition]<br>Set when TCNT overflows (changed from H'FF to H'00) during watchdog timer operation |

Note: * Can only be written with 0 for flag clearing.

The method for writing to RSTCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 13.2.4, Notes on Register Access.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Counter Start

| | |
|---|---|
| 0 | $TCNT_n$ count operation is stopped |
| 1 | $TCNT_n$ performs count operation |

(n = 5 to 0)

Note: If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.

**TSYR—Timer Synchro Register** **H'FFC1** **TPU**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Synchronization

| | |
|---|---|
| 0 | $TCNT_n$ operates independently (TCNT presetting/ clearing is unrelated to other channels) |
| 1 | $TCNT_n$ performs synchronous operation<br>TCNT synchronous presetting/synchronous clearing is possible |

(n = 5 to 0)

Notes: 1. To set synchronous operation, the SYNC bits for at least two channels must be set to 1.
2. To set synchronous clearing, in addition to the SYNC bit , the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | External clock: counts on TCLKC pin input |
| | | 1 | External clock: counts on TCLKD pin input |

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Counter Clear

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRA compare match/input capture |
| | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*1 |
| 1 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRC compare match/input capture*2 |
| | 1 | 0 | TCNT cleared by TGRD compare match/input capture*2 |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | BFB | BFA | MD3*1 | MD2*2 | MD1 | MD0 |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

TGRA Buffer Operation

| | |
|---|---|
| 0 | TGRA operates normally |
| 1 | TGRA and TGRC used together for buffer operation |

TGRB Buffer Operation

| | |
|---|---|
| 0 | TGRB operates normally |
| 1 | TGRB and TGRD used together for buffer operation |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR0A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0A is input capture register | Capture input source is TIOCA0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down |

* : Don't care

TGR0B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0B is input capture register | Capture input source is TIOCB0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down*1 |

* : Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000, and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | : | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR0C I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0C is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0C is input capture register | Capture input source is TIOCC0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down |

* : Don't care

Note: When the BFA bit in TMDR0 is set to 1 and TGR0C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

TGR0D I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0D is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0D is input capture registe*2 | Capture input source is TIOCD0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down*1 |

* : Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000, and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.
2. When the BFB bit in TMDR0 is set to 1 and TGR0D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Note: When GRC or GRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

TGR Interrupt Enable C

| | |
|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled |

TGR Interrupt Enable D

| | |
|---|---|
| 0 | Interrupt requests (TGID) by TGFD bit disabled |
| 1 | Interrupt requests (TGID) by TGFD bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Input Capture/Output Compare Flag C

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

Input Capture/Output Compare Flag D

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Note: * Can only be written with 0 for flag clearing.

**TCNT0—Timer Counter 0** **H'FFD6** **TPU0**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up-counter

**TGR0A—Timer General Register 0A** **H'FFD8** **TPU0**
**TGR0B—Timer General Register 0B** **H'FFDA** **TPU0**
**TGR0C—Timer General Register 0C** **H'FFDC** **TPU0**
**TGR0D—Timer General Register 0D** **H'FFDE** **TPU0**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | Counts on TCNT2 overflow/underflow |

Note: This setting is ignored when channel 1 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | * | Count at both edges |

* : Don't care

Note: This setting is ignored when channel 1 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit in TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | 1Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Note: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR1A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR1A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR1A is input capture register | Capture input source is TIOCA1 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR0A compare match/ input capture | Input capture at generation of channel 0/TGR0A compare match/ input capture |

* : Don't care

TGR1B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR1B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR1B is input capture register | Capture input source is TIOCB1 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR0C compare match/ input capture | Input capture at generation of TGR0B compare match/input capture |

* : Don't care

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGI Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT1—Timer Counter 1** **H'FFE6** **TPU1**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR1A—Timer General Register 1A** **H'FFE8** **TPU1**
**TGR1B—Timer General Register 1B** **H'FFEA** **TPU1**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | External clock: counts on TCLKC pin input |
| | | 1 | Internal clock: counts on ø/1024 |

Note: This setting is ignored when channel 2 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | * | Count at both edges |

* : Don't care

Note: This setting is ignored when channel 2 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Note: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR2A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR2A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR2A is input capture register | Capture input source is TIOCA2 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

TGR2B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR2B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR2B is input capture register | Capture input source is TIOCB2 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT2—Timer Counter 2** **H'FFF6** **TPU2**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR2A—Timer General Register 2A** **H'FFF8** **TPU2**
**TGR2B—Timer General Register 2B** **H'FFFA** **TPU2**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

## C.1 Port 1 Block Diagram

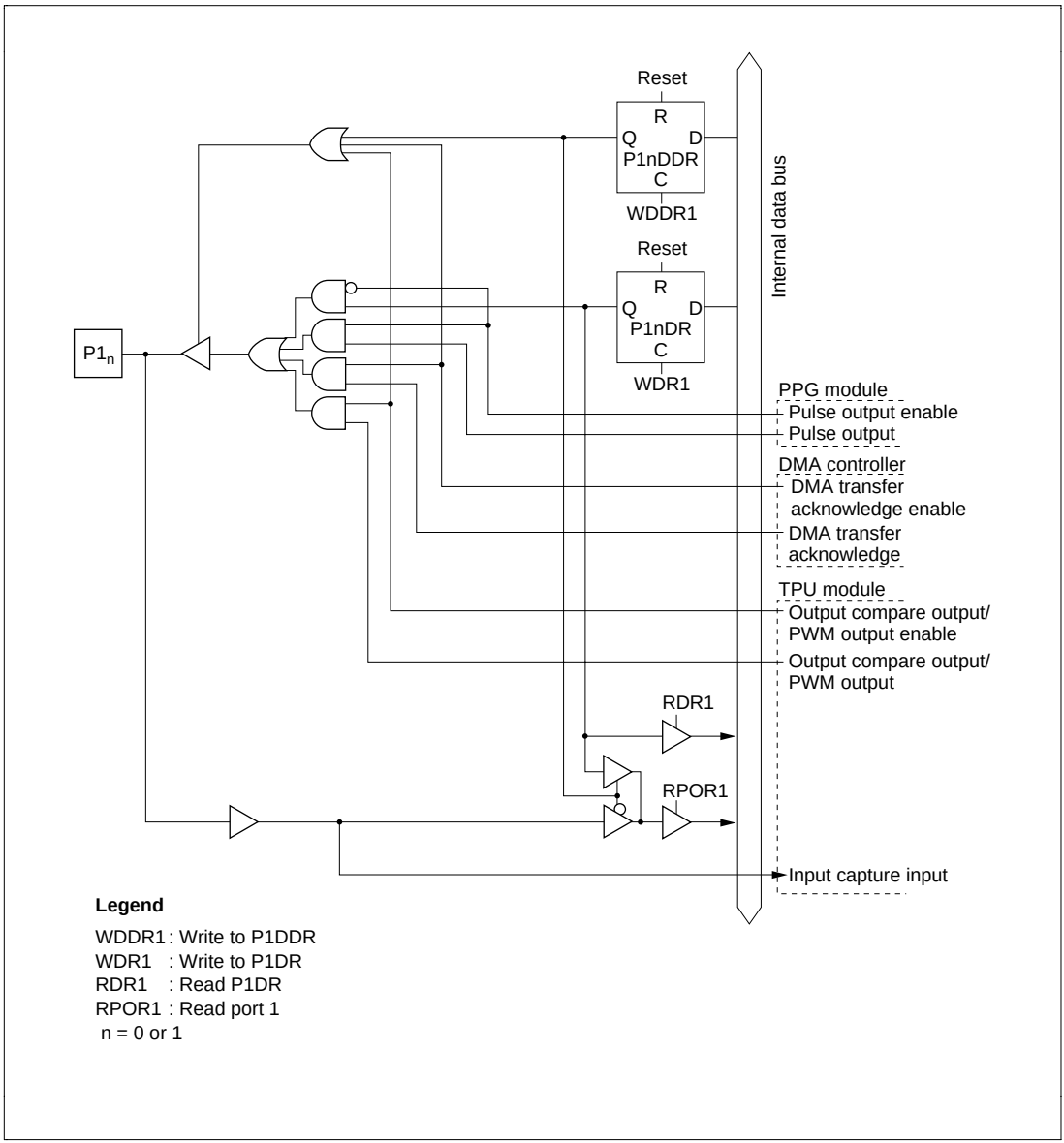



**Figure C-1 (a) Port 1 Block Diagram (Pins $P1_0$ and $P1_1$)**

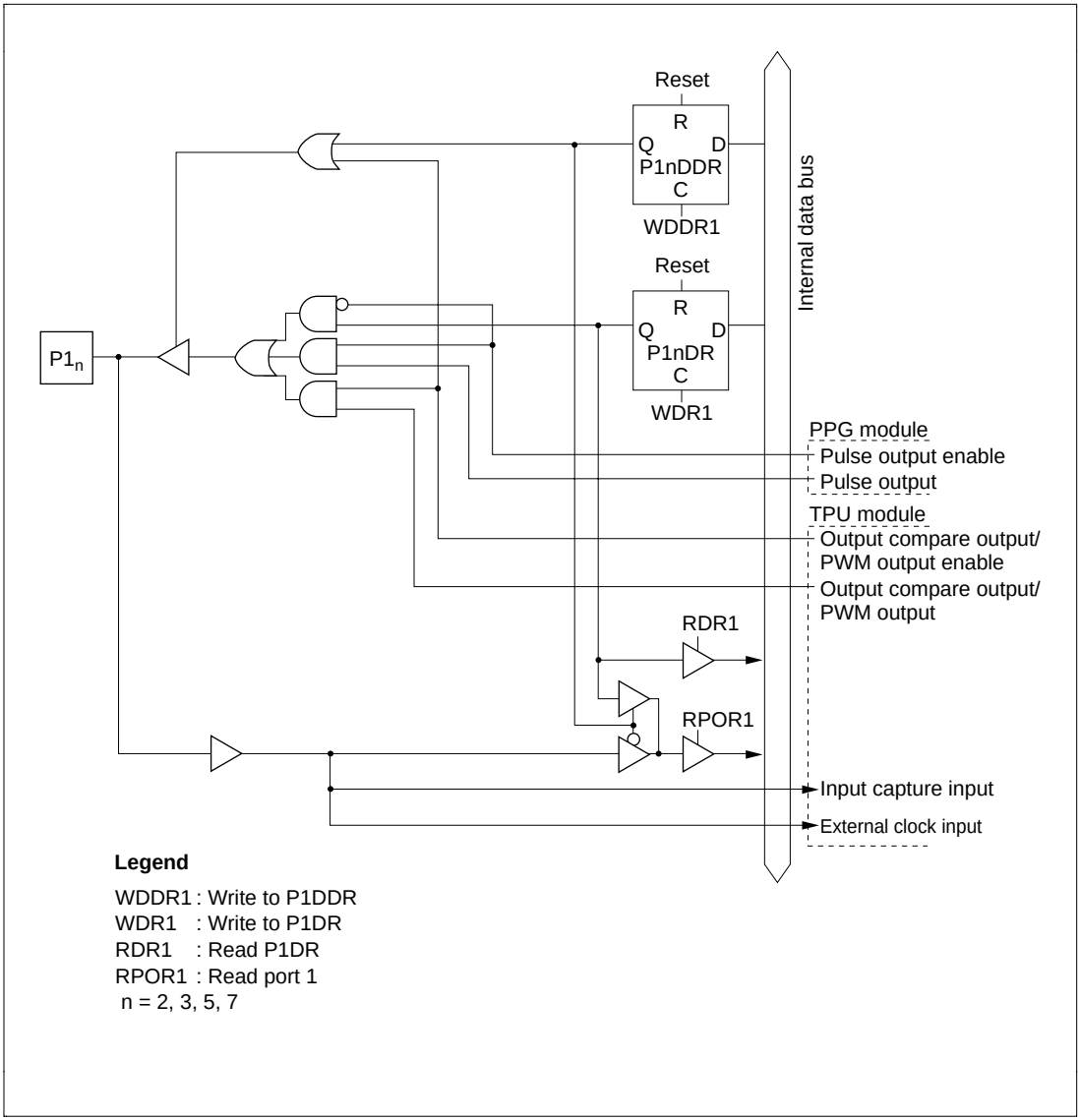



**Figure C-1 (b) Port 1 Block Diagram (Pins $P1_2$, $P1_3$, $P1_5$, and $P1_7$)**

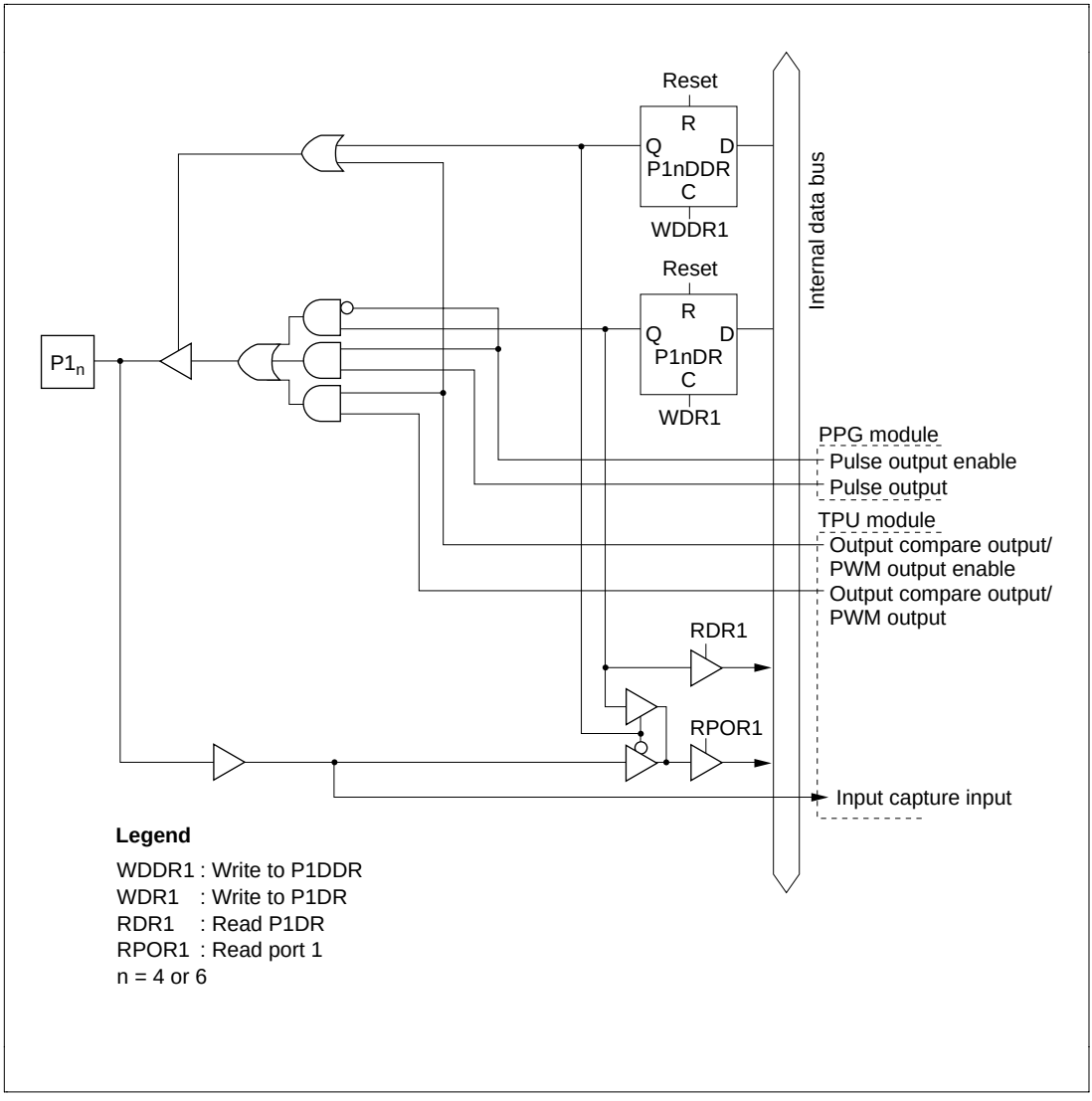



**Figure C-1 (c) Port 1 Block Diagram (Pins $P1_4$ and $P1_6$)**

## C.2 Port 2 Block Diagram

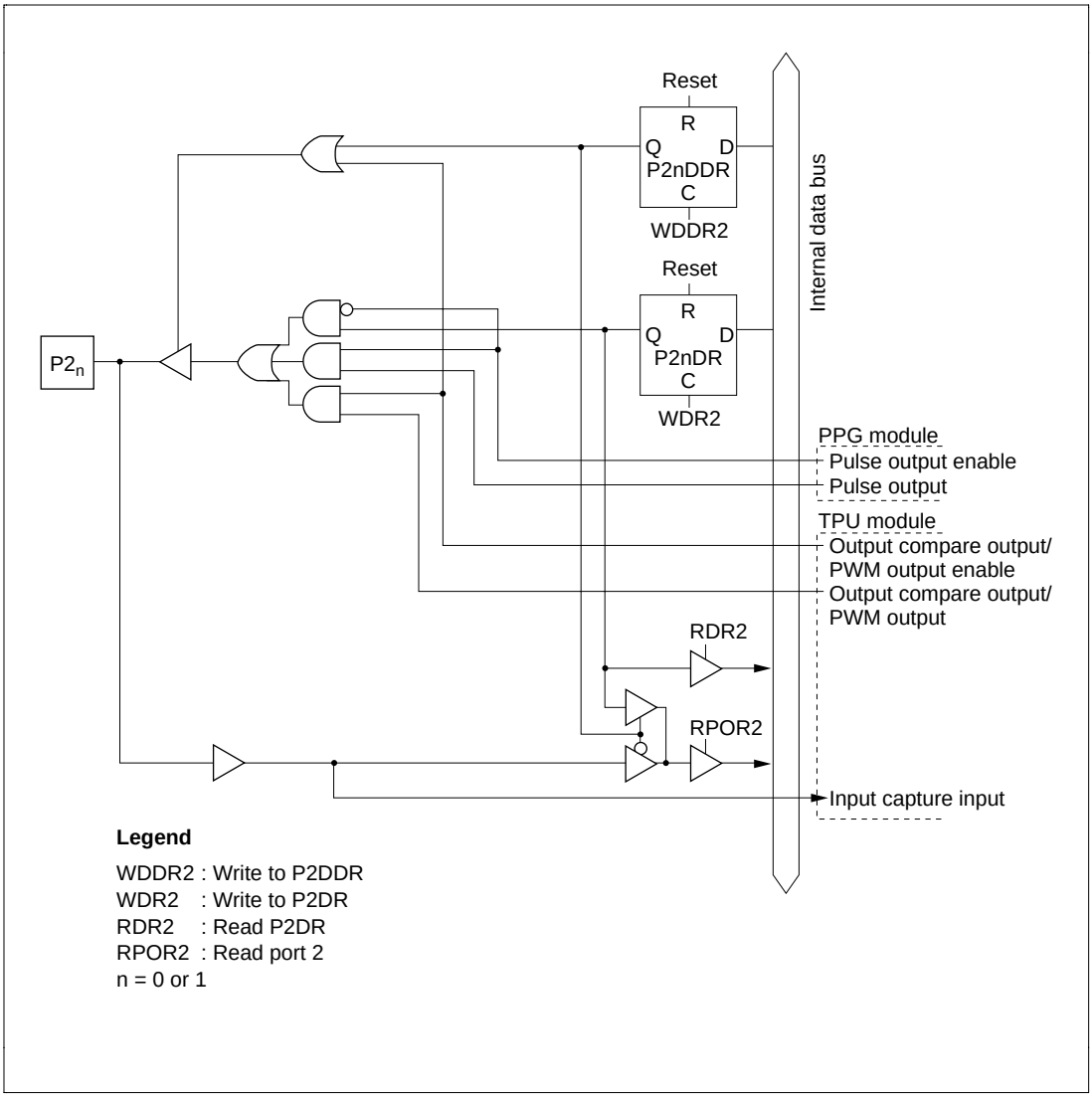



**Figure C-2 (a) Port 2 Block Diagram (Pins $P2_0$ and $P2_1$)**

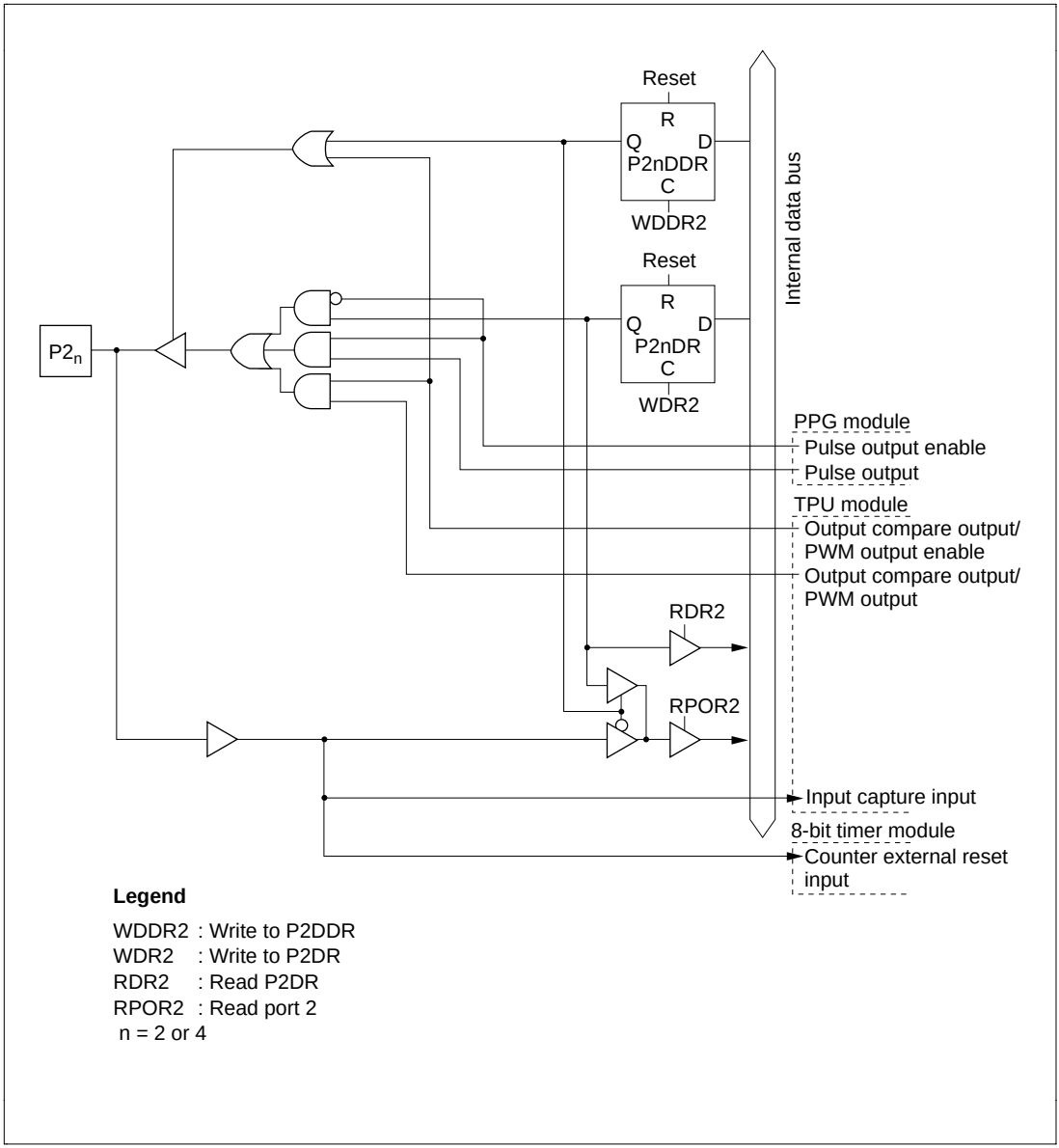



**Figure C-2 (b) Port 2 Block Diagram (Pins $P2_2$ and $P2_4$)**

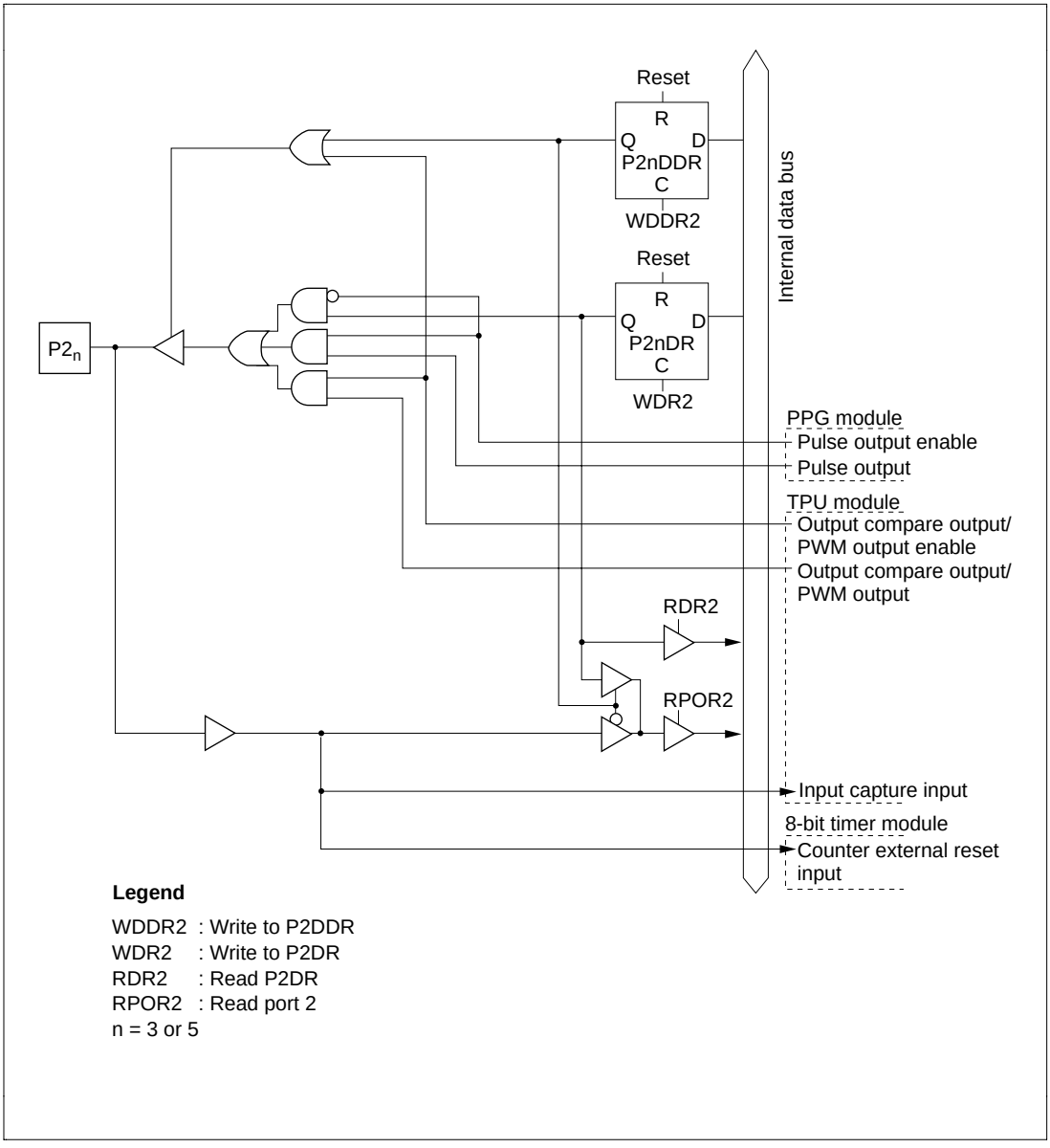



**Figure C-2 (c) Port 2 Block Diagram (Pins $P2_3$ and $P2_5$)**

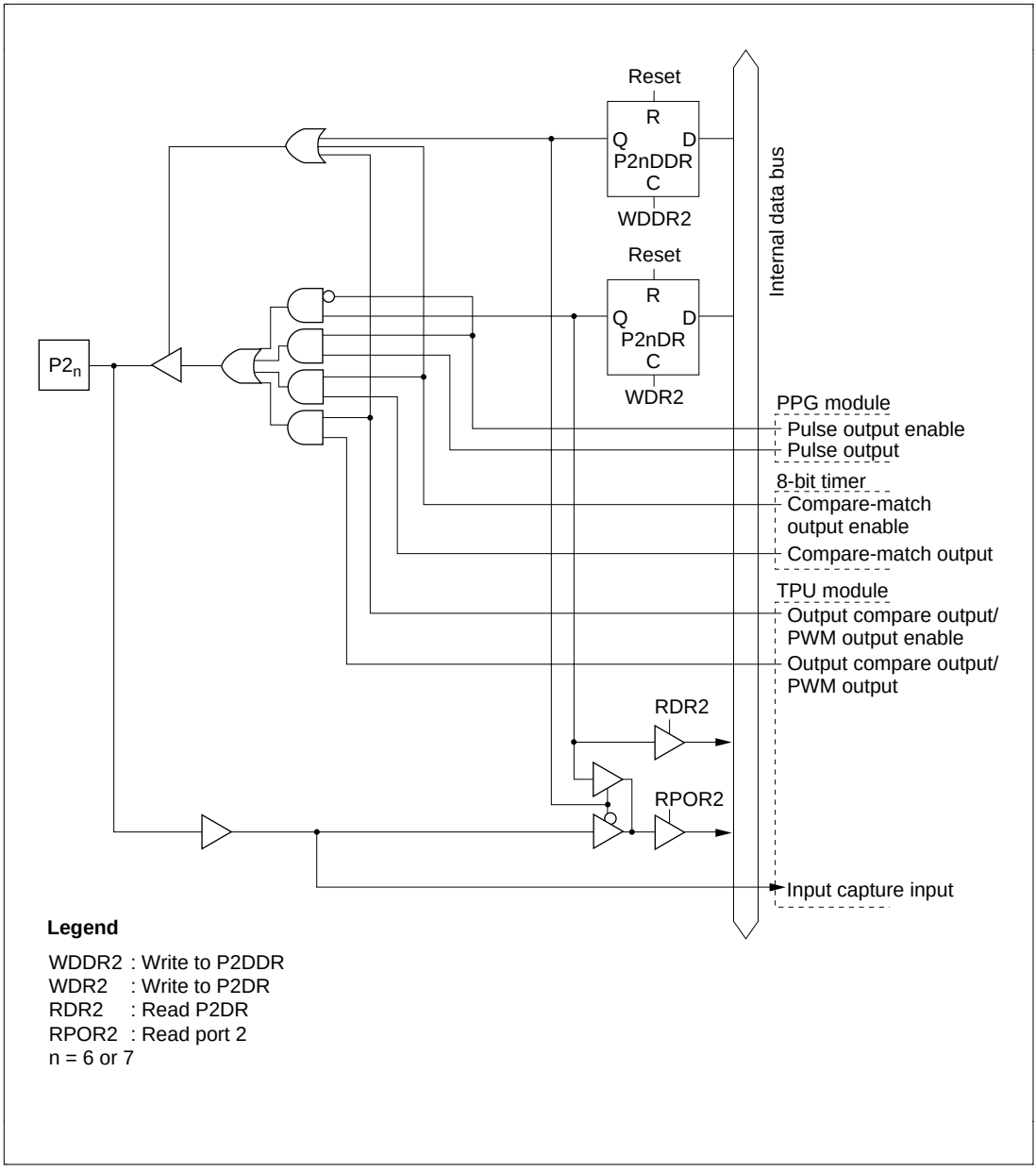
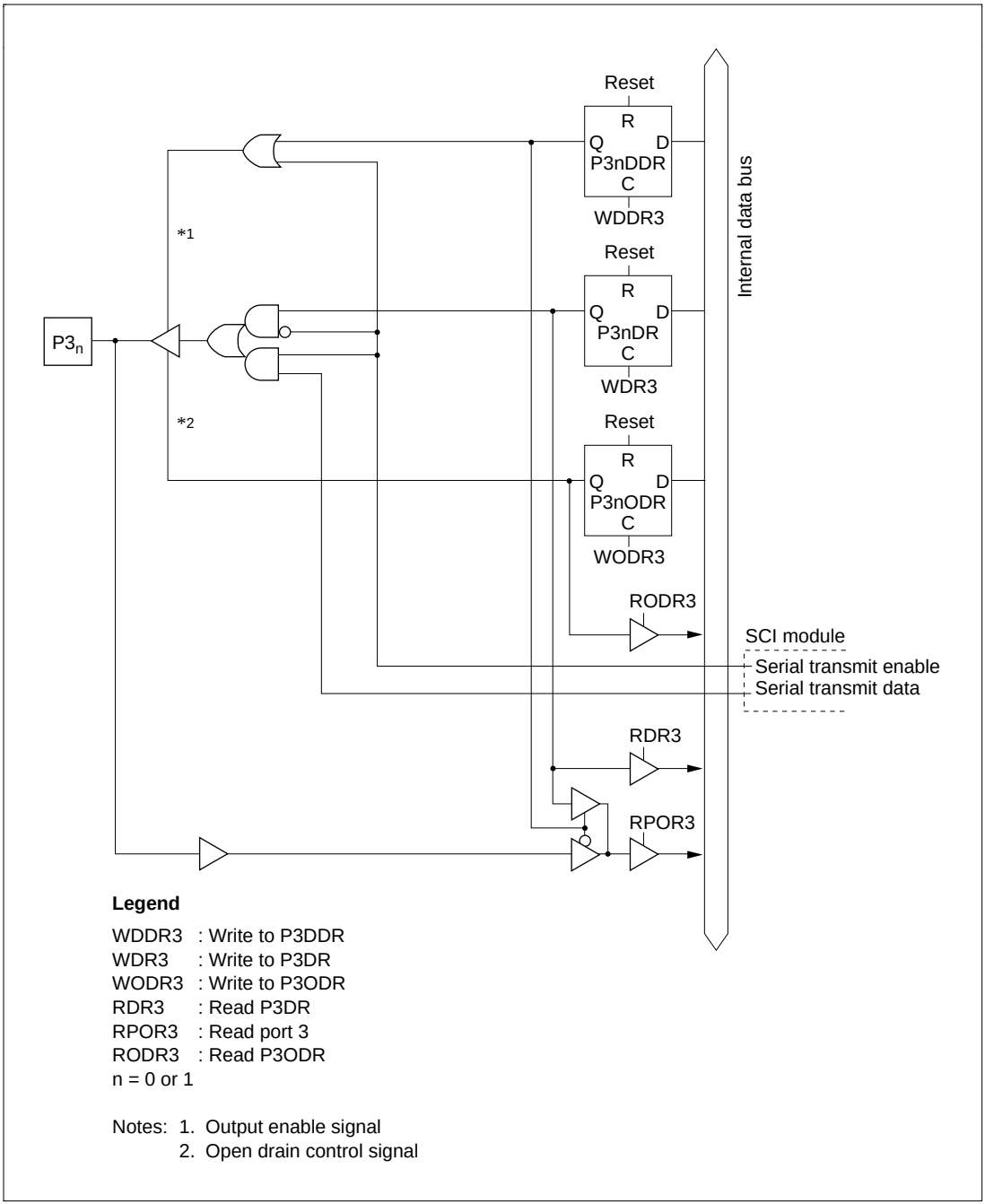



**Figure C-2 (d) Port 2 Block Diagram (Pins $P2_6$ and $P2_7$)**

## C.3 Port 3 Block Diagram

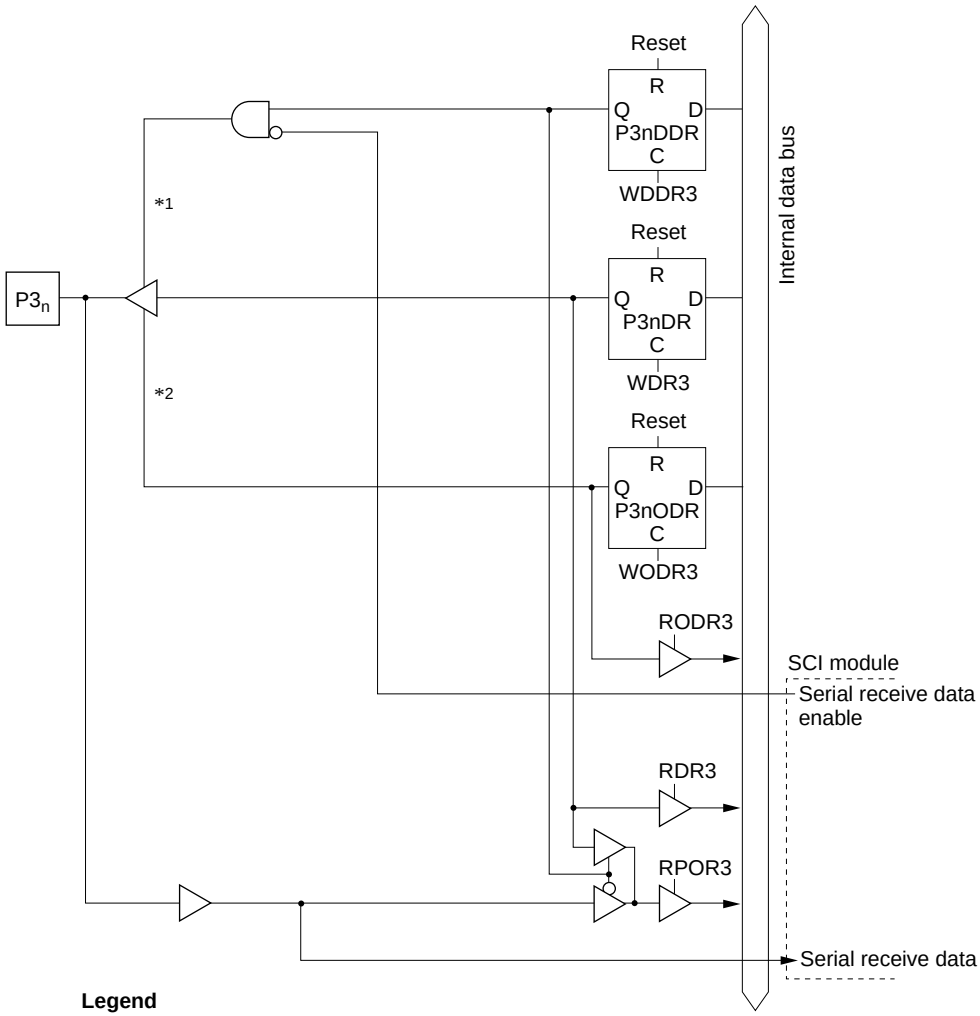



**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 0 or 1

Notes: 1. Output enable signal
2. Open drain control signal

**Figure C-3 (a) Port 3 Block Diagram (Pins $P3_0$ and $P3_1$)**

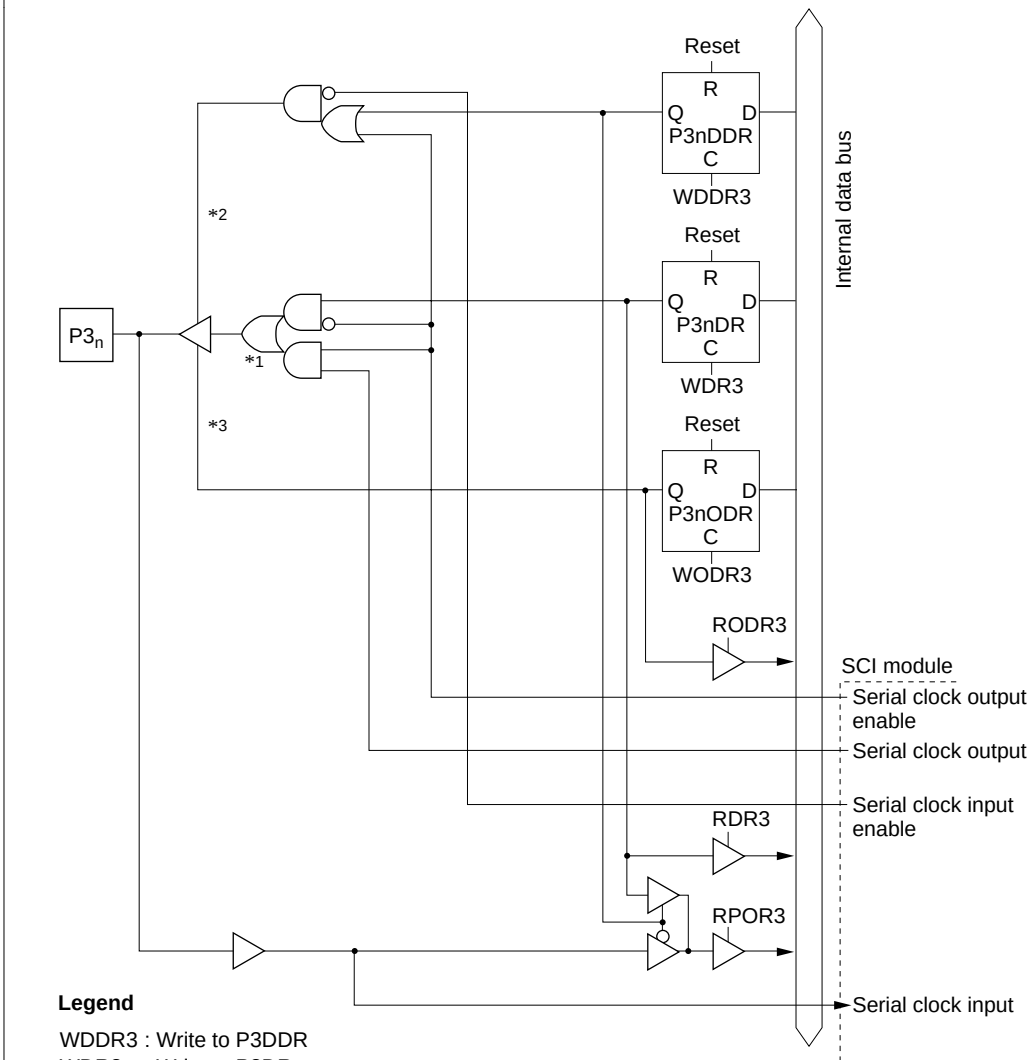



**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 2 or 3

Notes: 1. Output enable signal
2. Open drain control signal

**Figure C-3 (b) Port 3 Block Diagram (Pins $P3_2$ and $P3_3$)**




**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 4 or 5

Notes: 1. Priority order: Serial clock input > serial clock output > DR output
2. Output enable signal
3. Open drain control signal

**Figure C-3 (c) Port 3 Block Diagram (Pins $P3_4$ and $P3_5$)**

## C.4 Port 4 Block Diagram

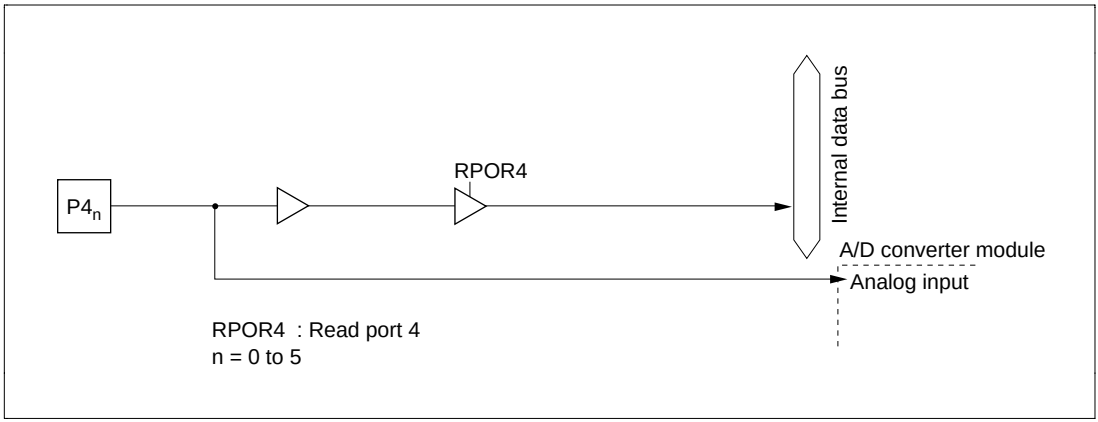



**Figure C-4 (a) Port 4 Block Diagram (Pins $P4_0$ to $P4_5$)**

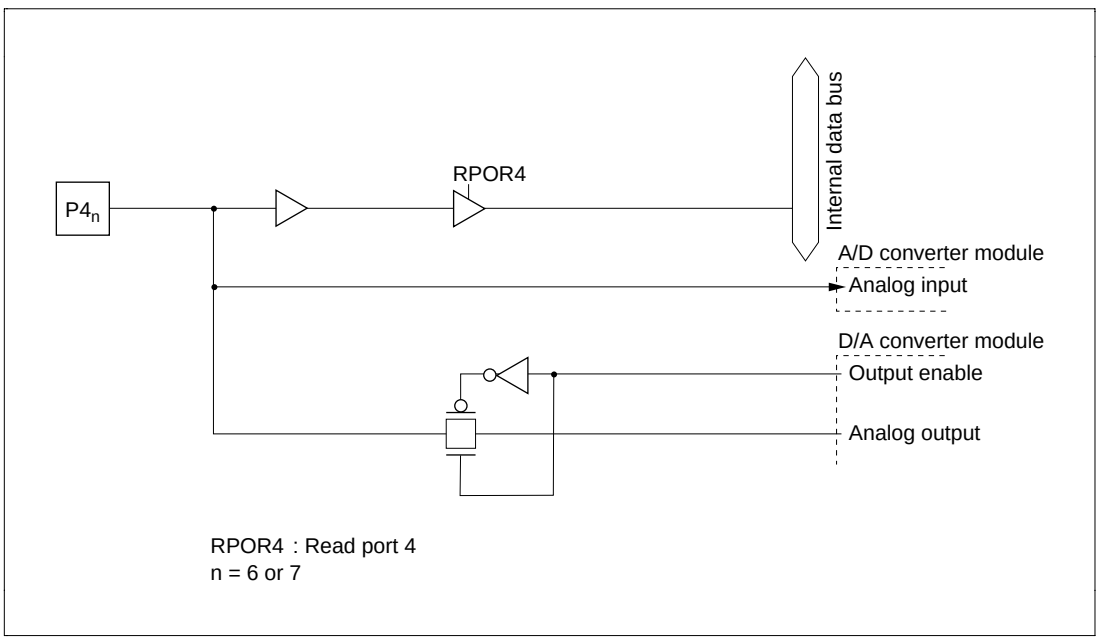



**Figure C-4 (b) Port 4 Block Diagram (Pins $P4_6$ and $P4_7$)**

## C.5 Port 5 Block Diagram

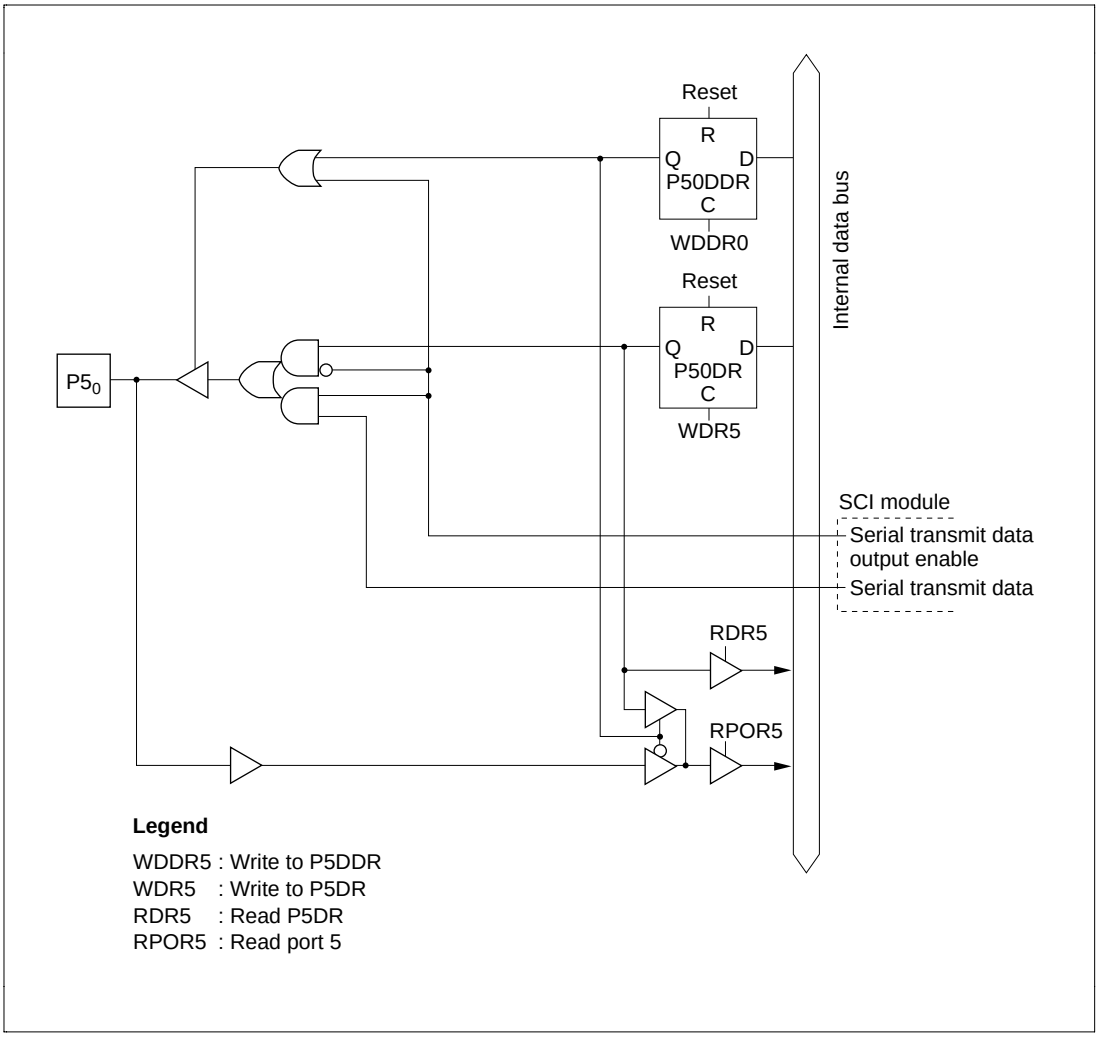



**Figure C-5 (a) Port 5 Block Diagram (Pin $P5_0$)**

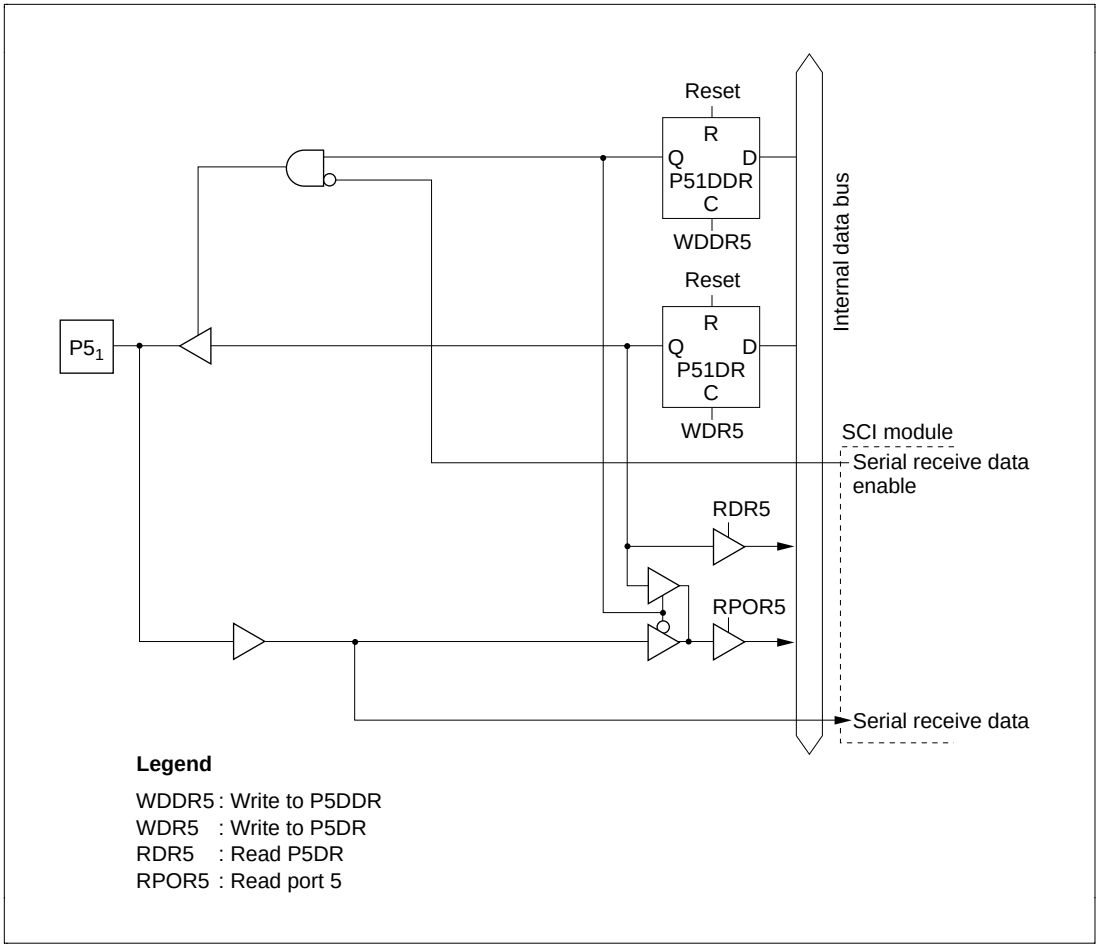



**Figure C-5 (b) Port 5 Block Diagram (Pin $P5_1$)**

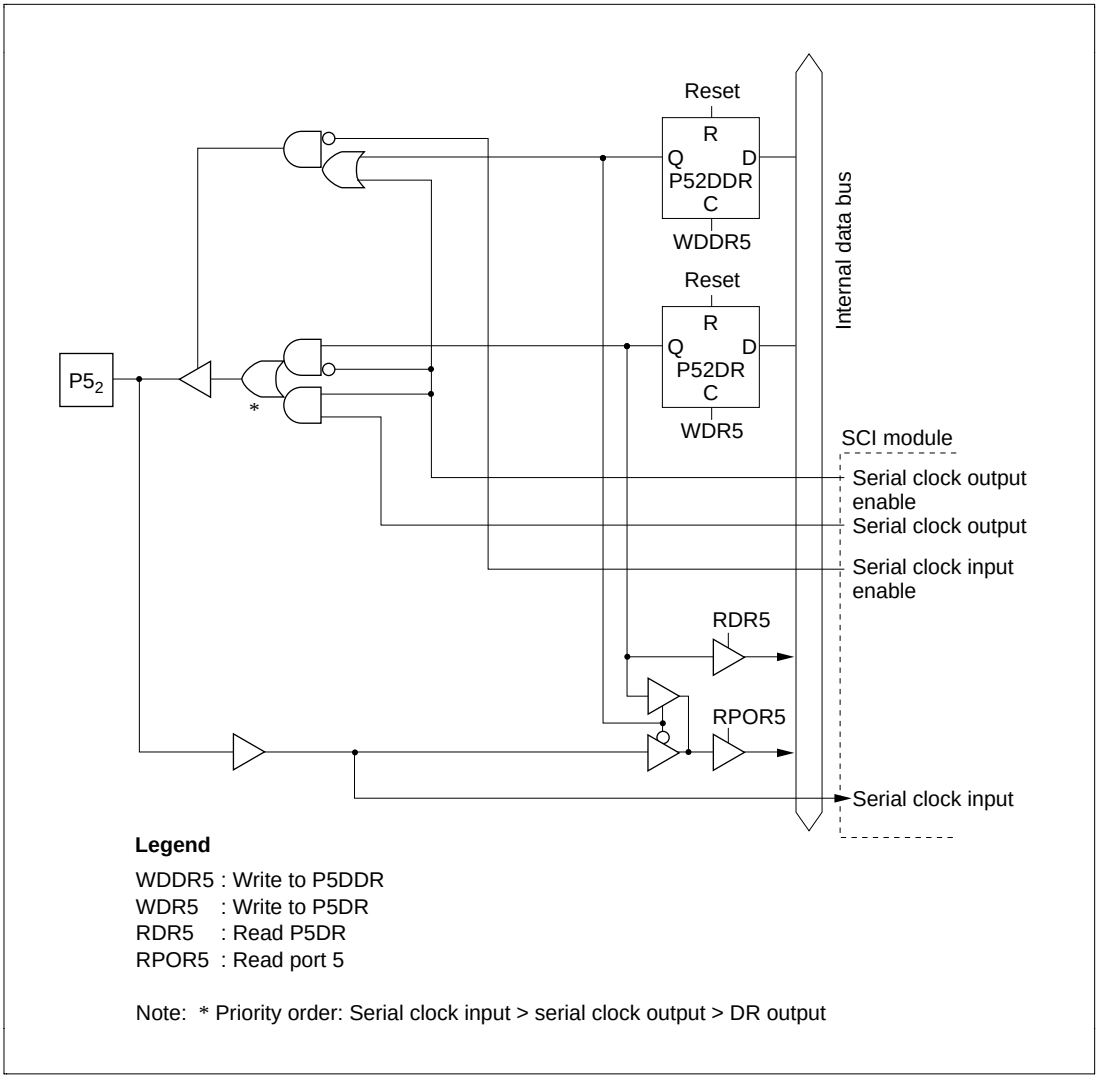



**Figure C-5 (c) Port 5 Block Diagram (Pin P5$_2$)**

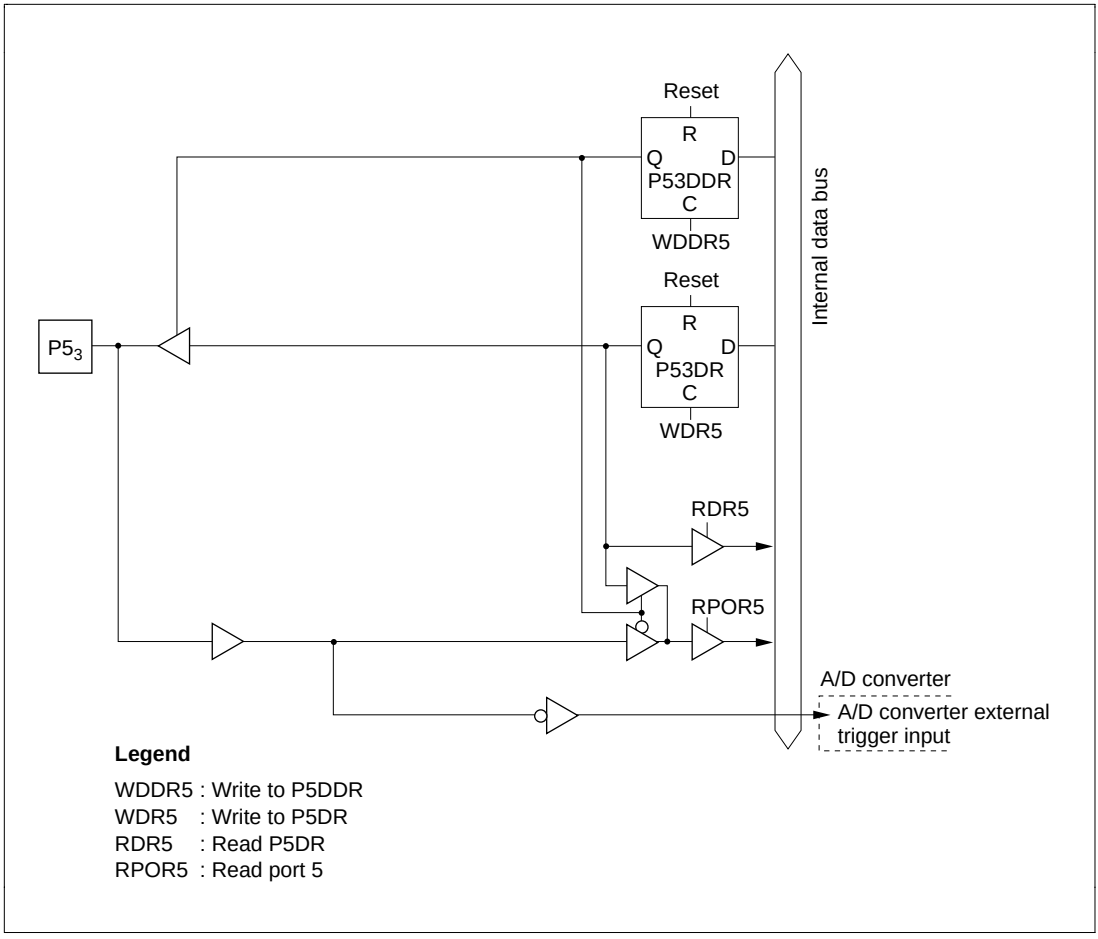



**Figure C-5 (d) Port 5 Block Diagram (Pin P5$_3$)**

## C.6 Port 6 Block Diagram

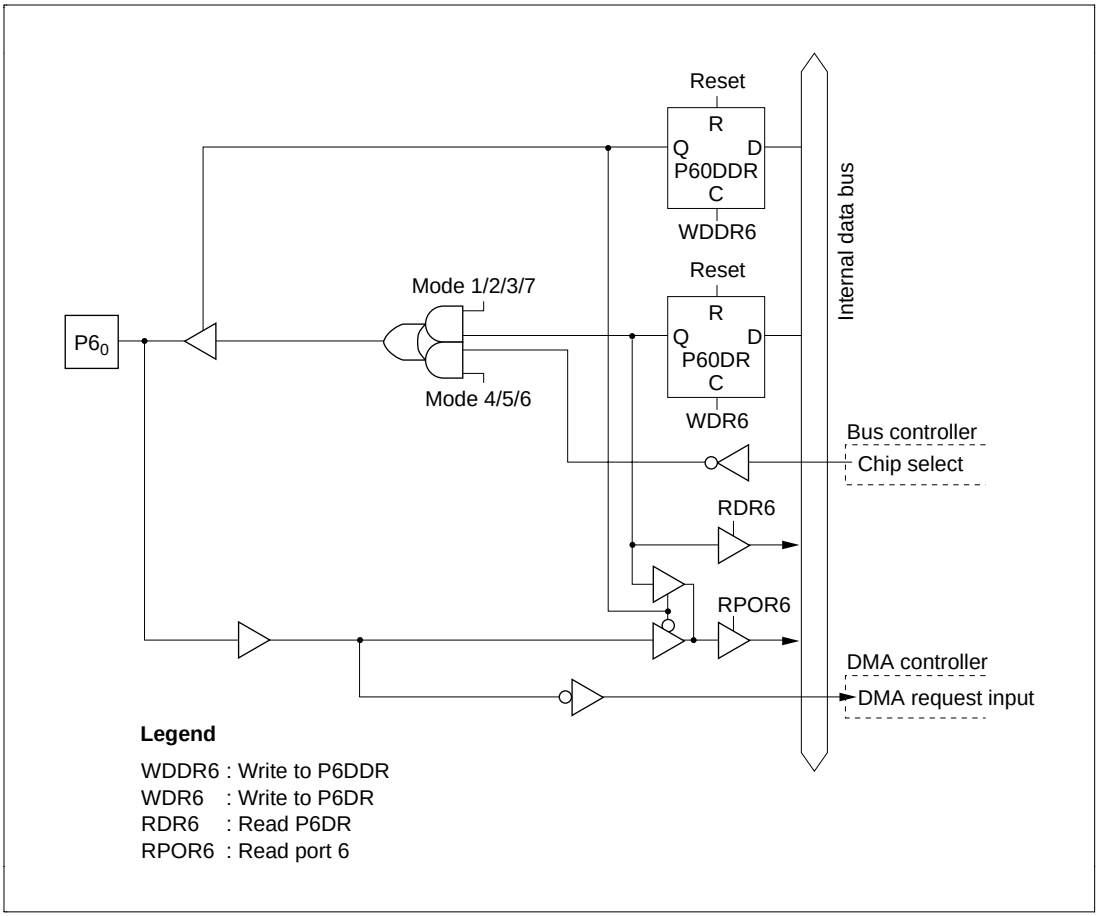



**Figure C-6 (a) Port 6 Block Diagram (Pin $P6_0$)**

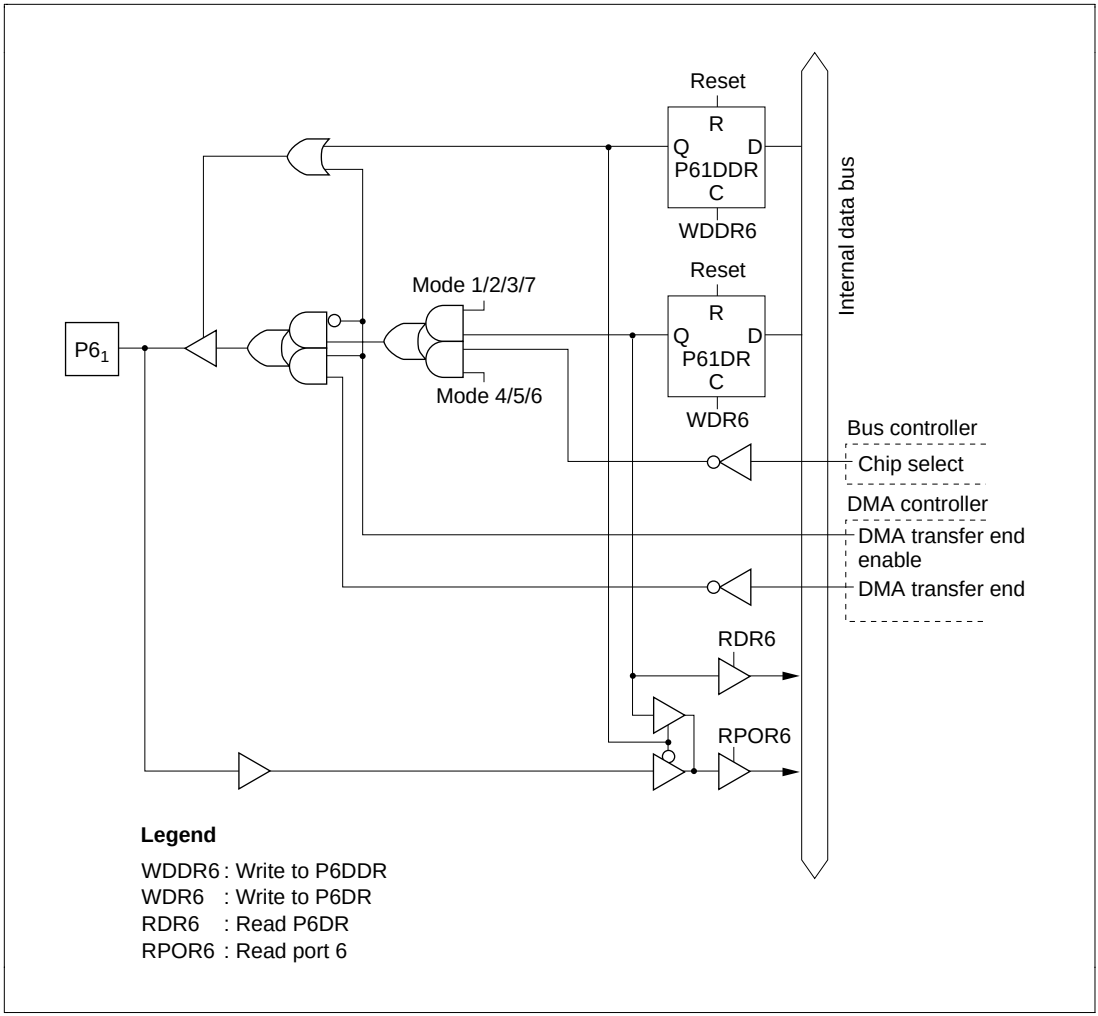



**Figure C-6 (b) Port 6 Block Diagram (Pin P6$_1$)**

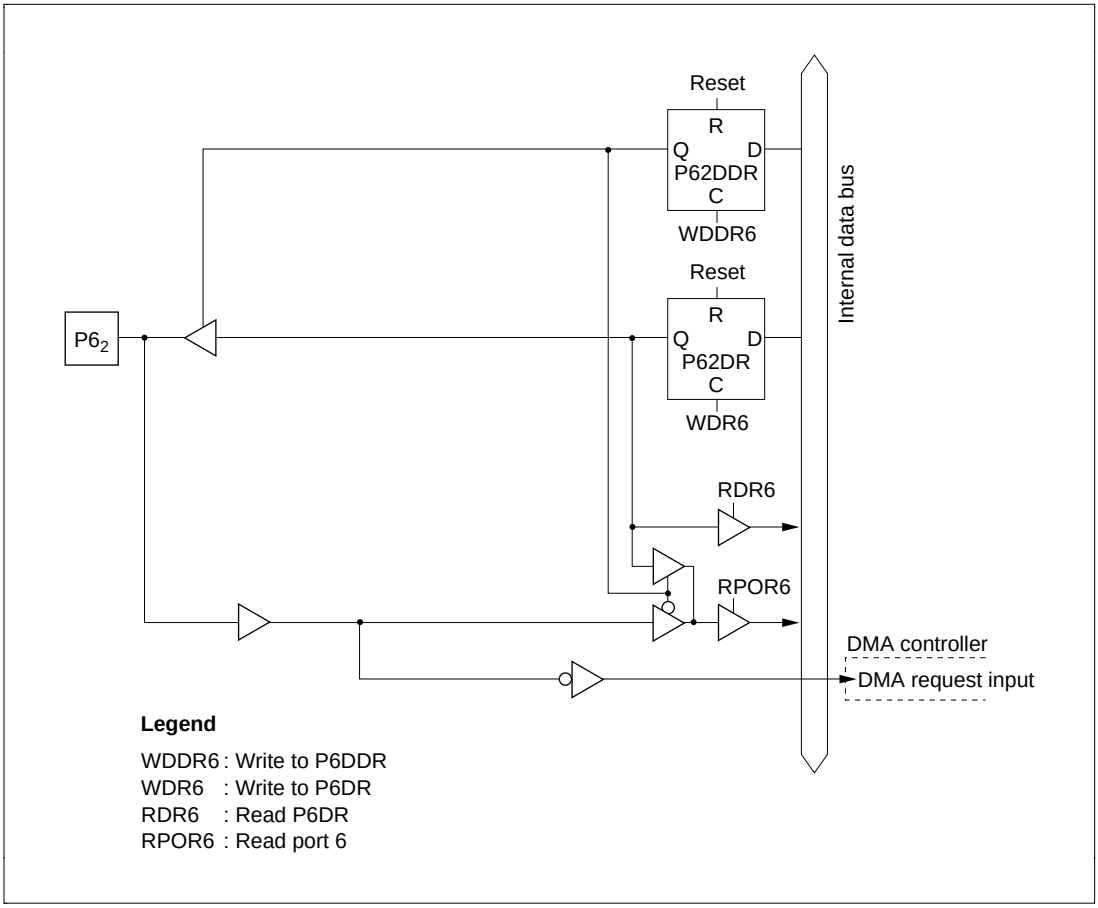



**Figure C-6 (c) Port 6 Block Diagram (Pin P6$_2$)**

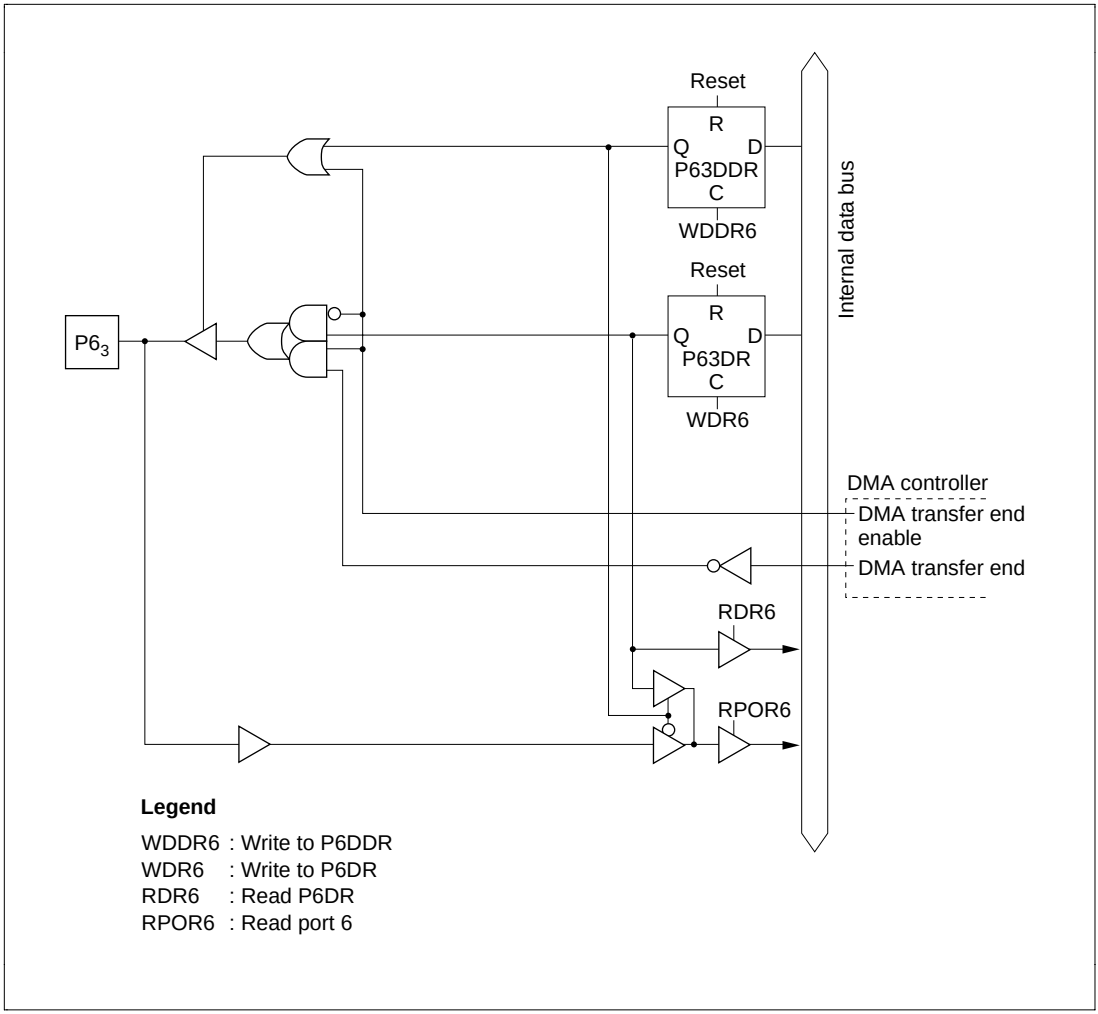



**Figure C-6 (d) Port 6 Block Diagram (Pin P6$_3$)**

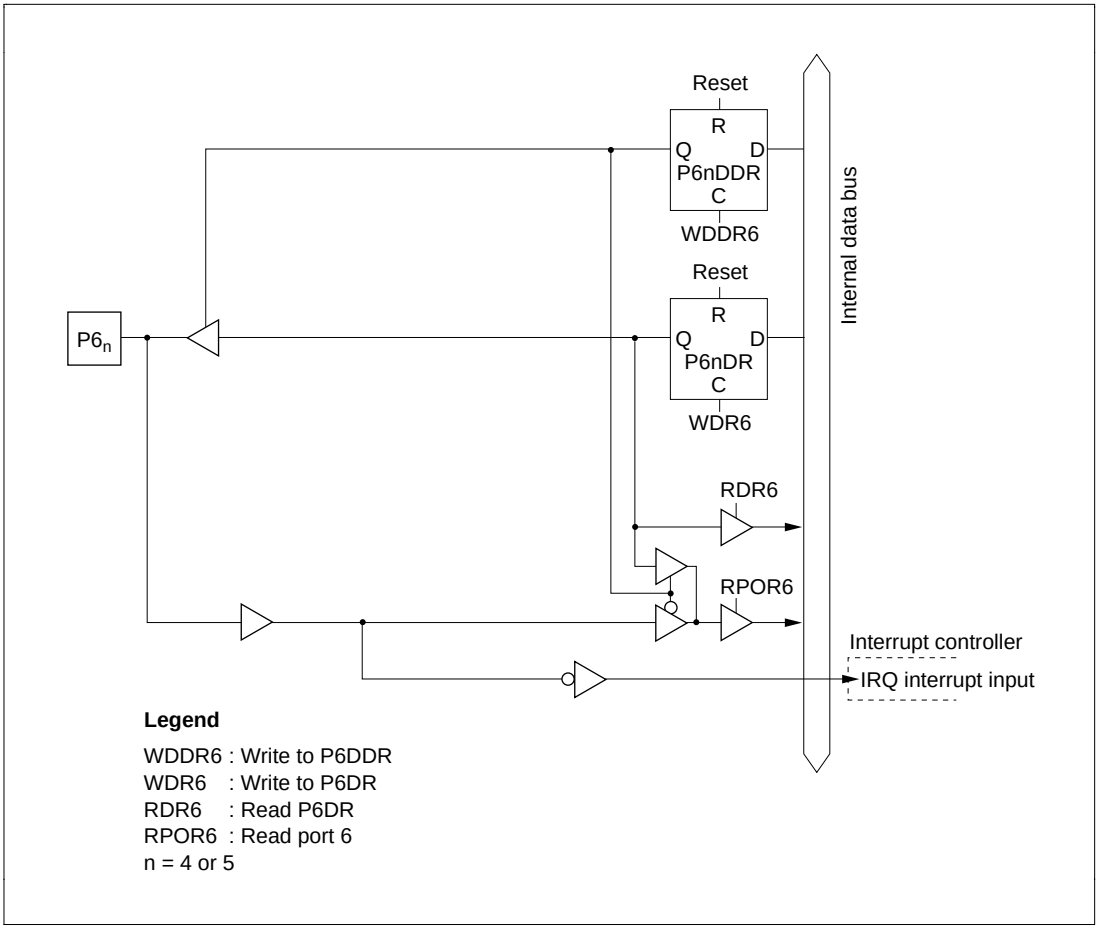



**Figure C-6 (e) Port 6 Block Diagram (Pins $P6_4$ and $P6_5$)**

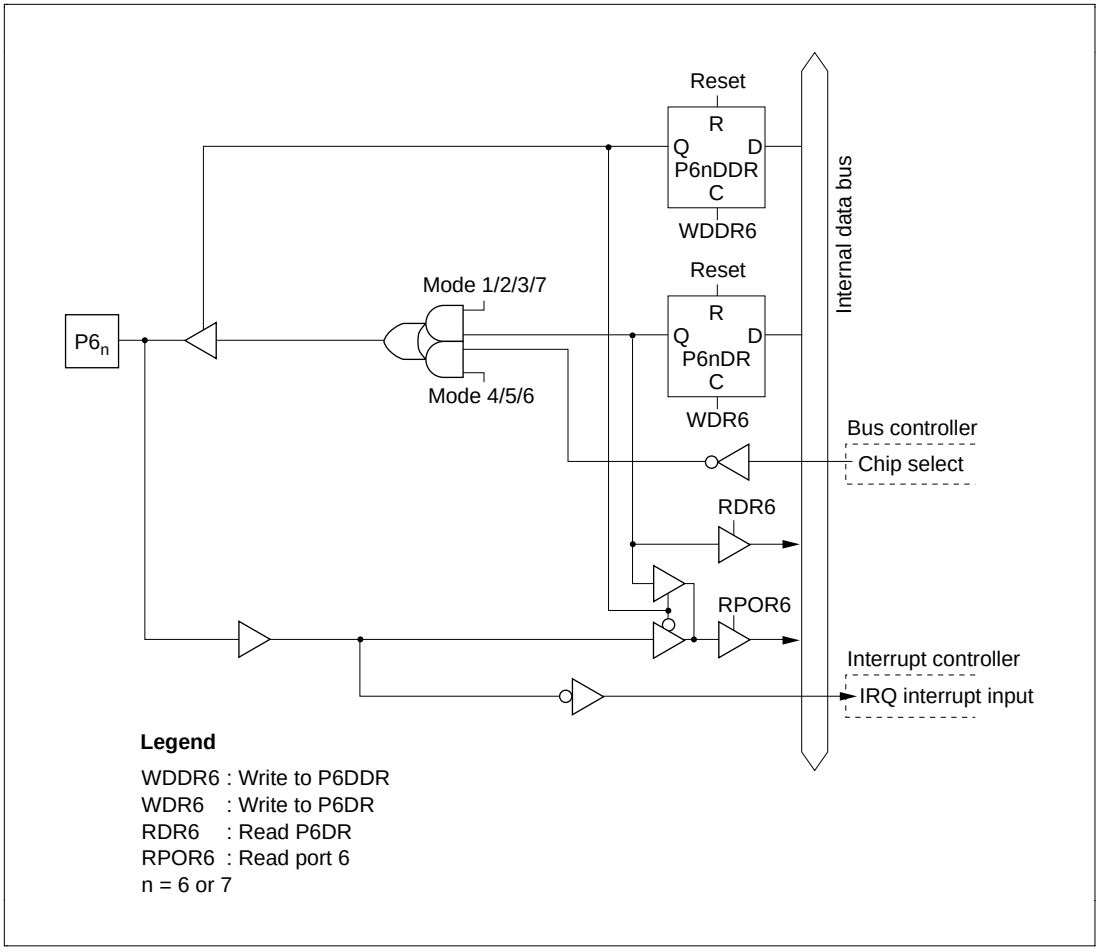



**Figure C-6 (f) Port 6 Block Diagram (Pins $P6_6$ and $P6_7$)**

## C.7 Port A Block Diagram

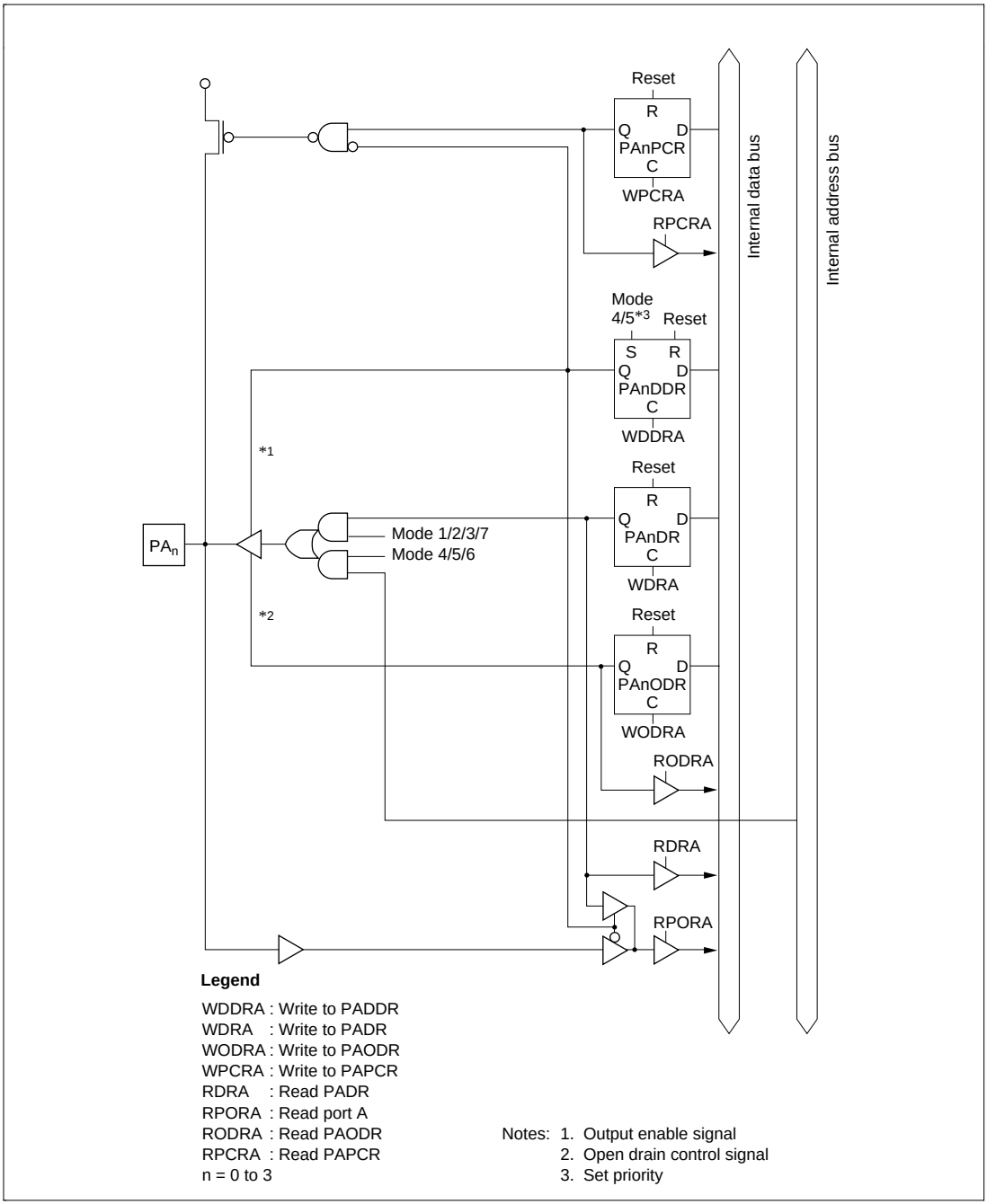



**Figure C-7 (a) Port A Block Diagram (Pins $PA_0$ to $PA_3$)**

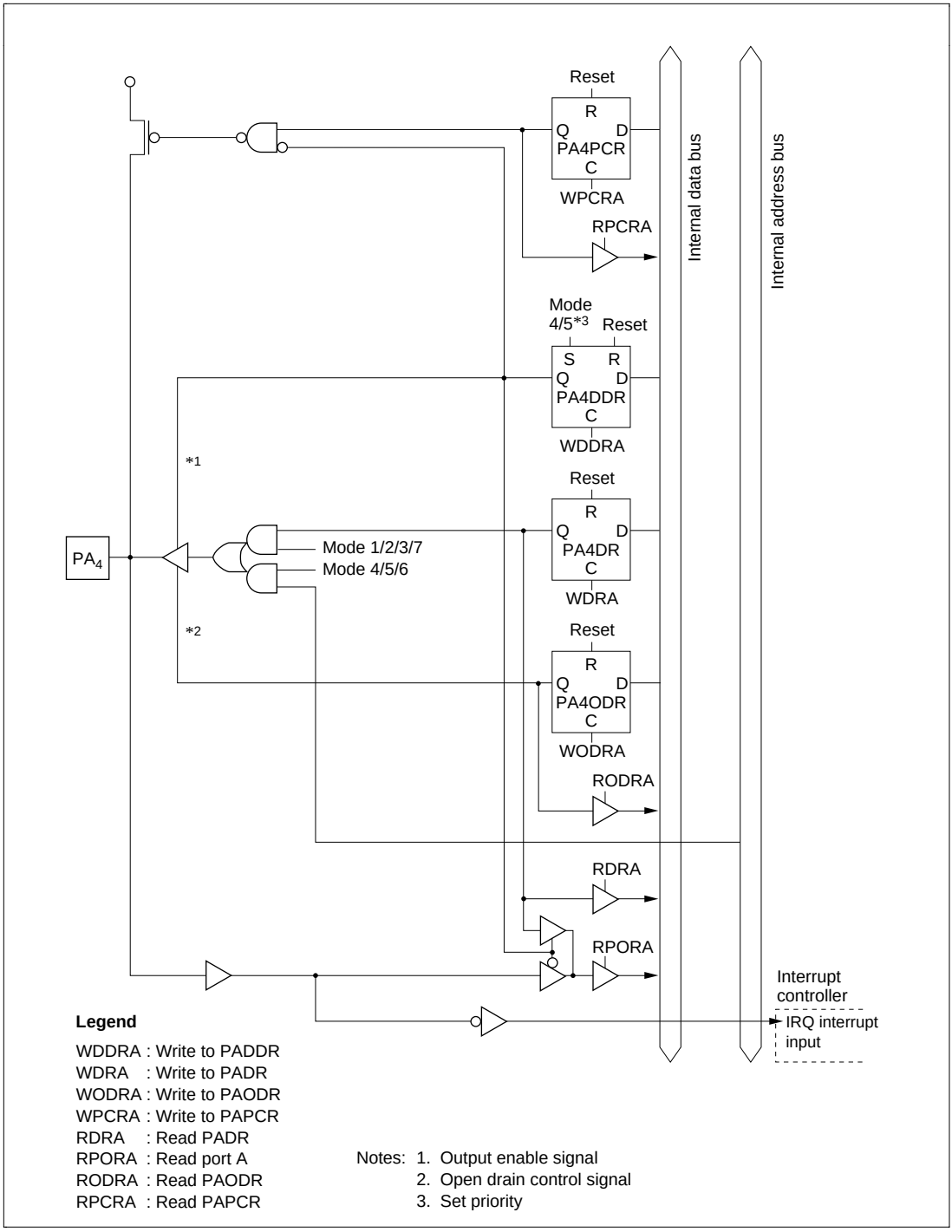



**Figure C-7 (b) Port A Block Diagram (Pin $PA_4$)**

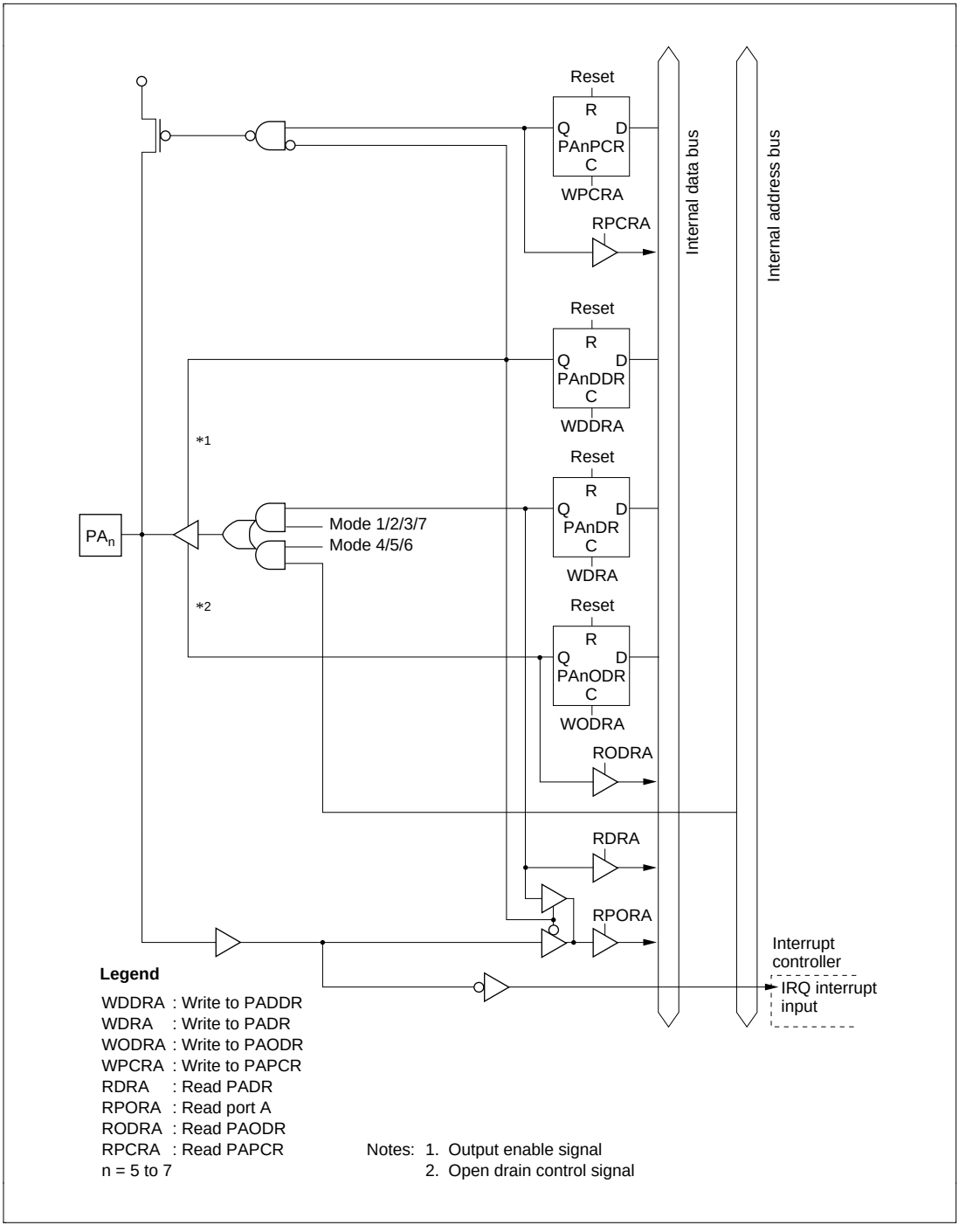



**Figure C-7 (c) Port A Block Diagram (Pins $PA_5$ to $PA_7$)**

## C.8 Port B Block Diagram

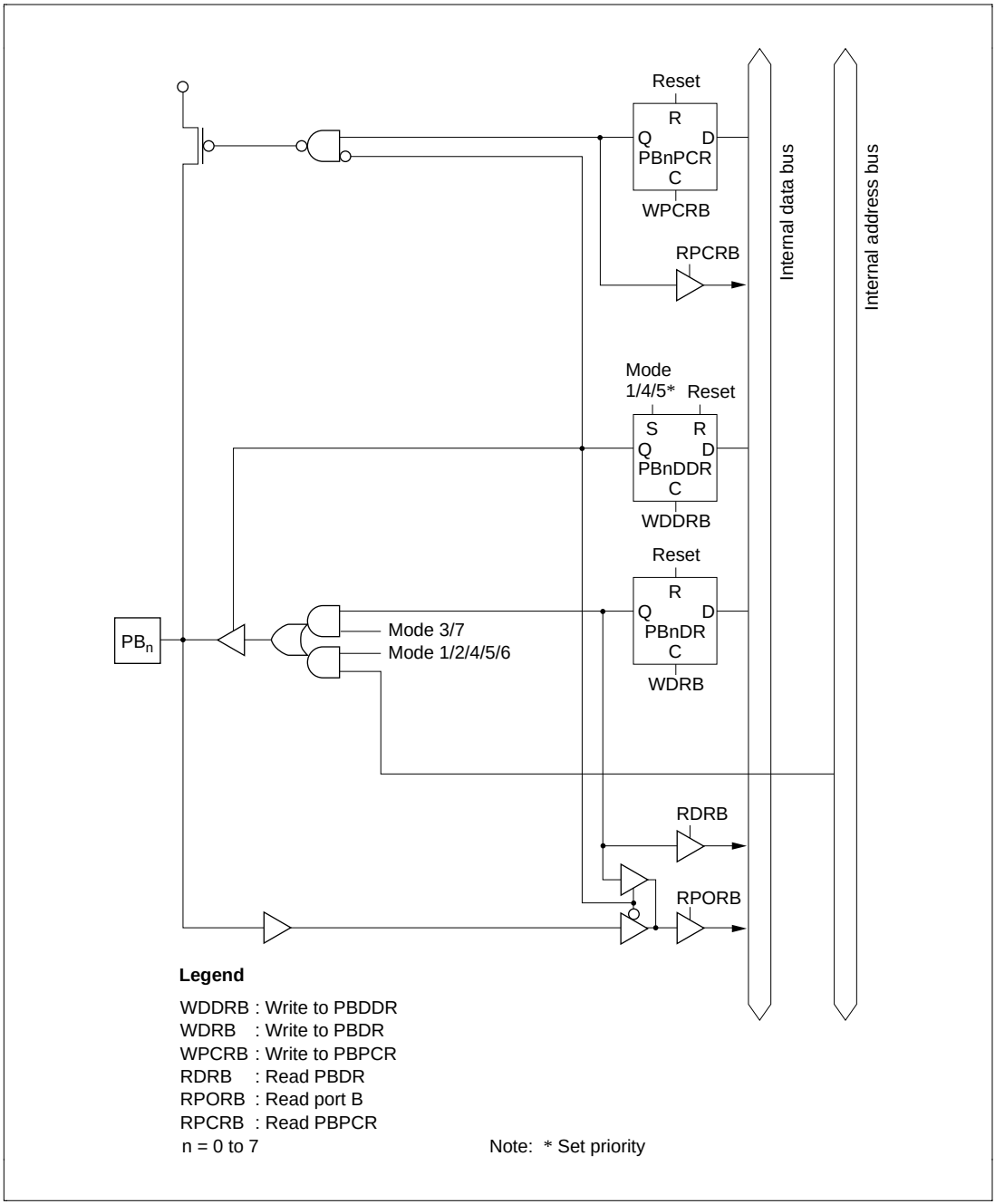



**Figure C-8 Port B Block Diagram (Pin $PB_n$)**

## C.9 Port C Block Diagram

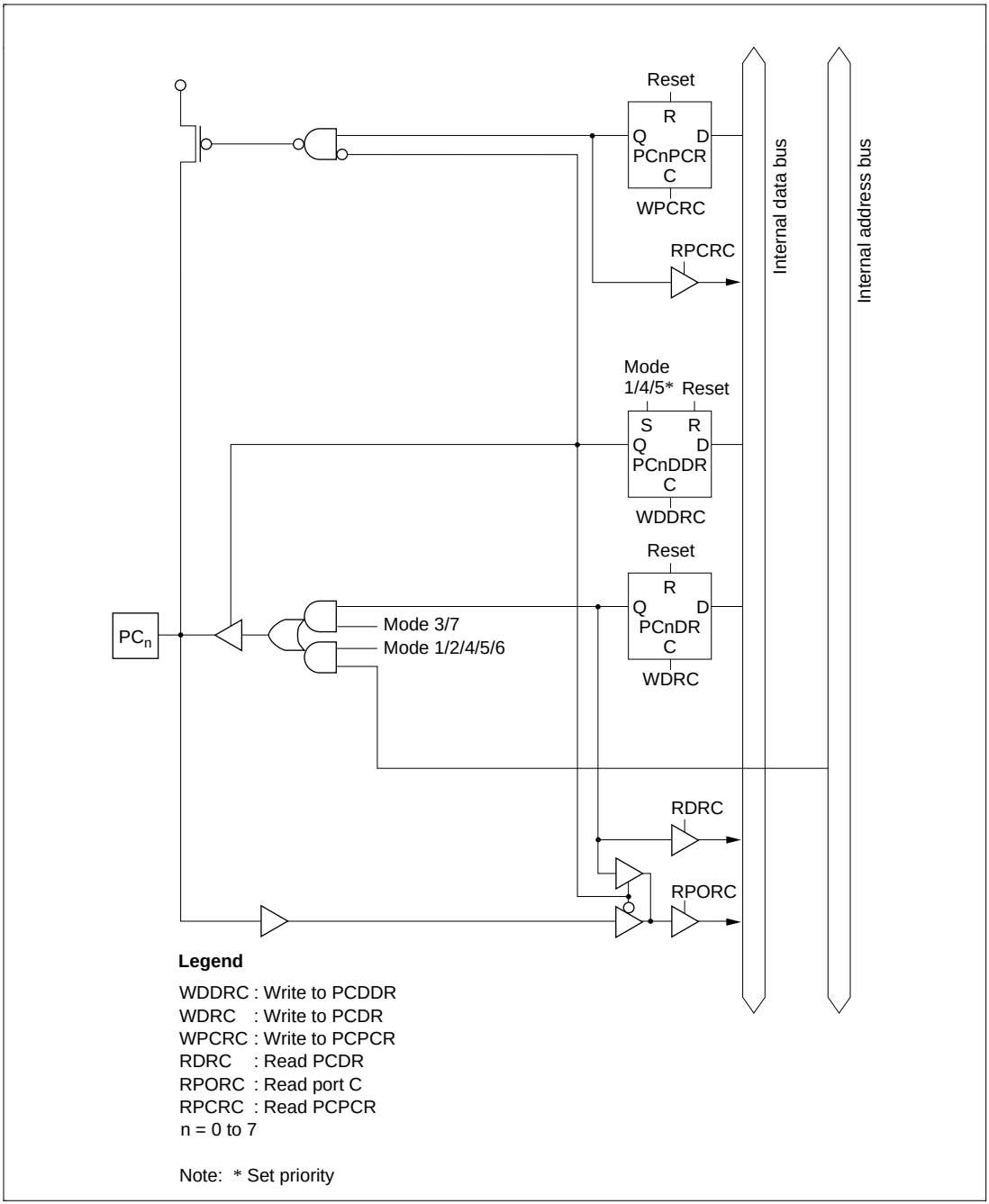



**Figure C-9 Port C Block Diagram (Pin $PC_n$)**

## C.10 Port D Block Diagram

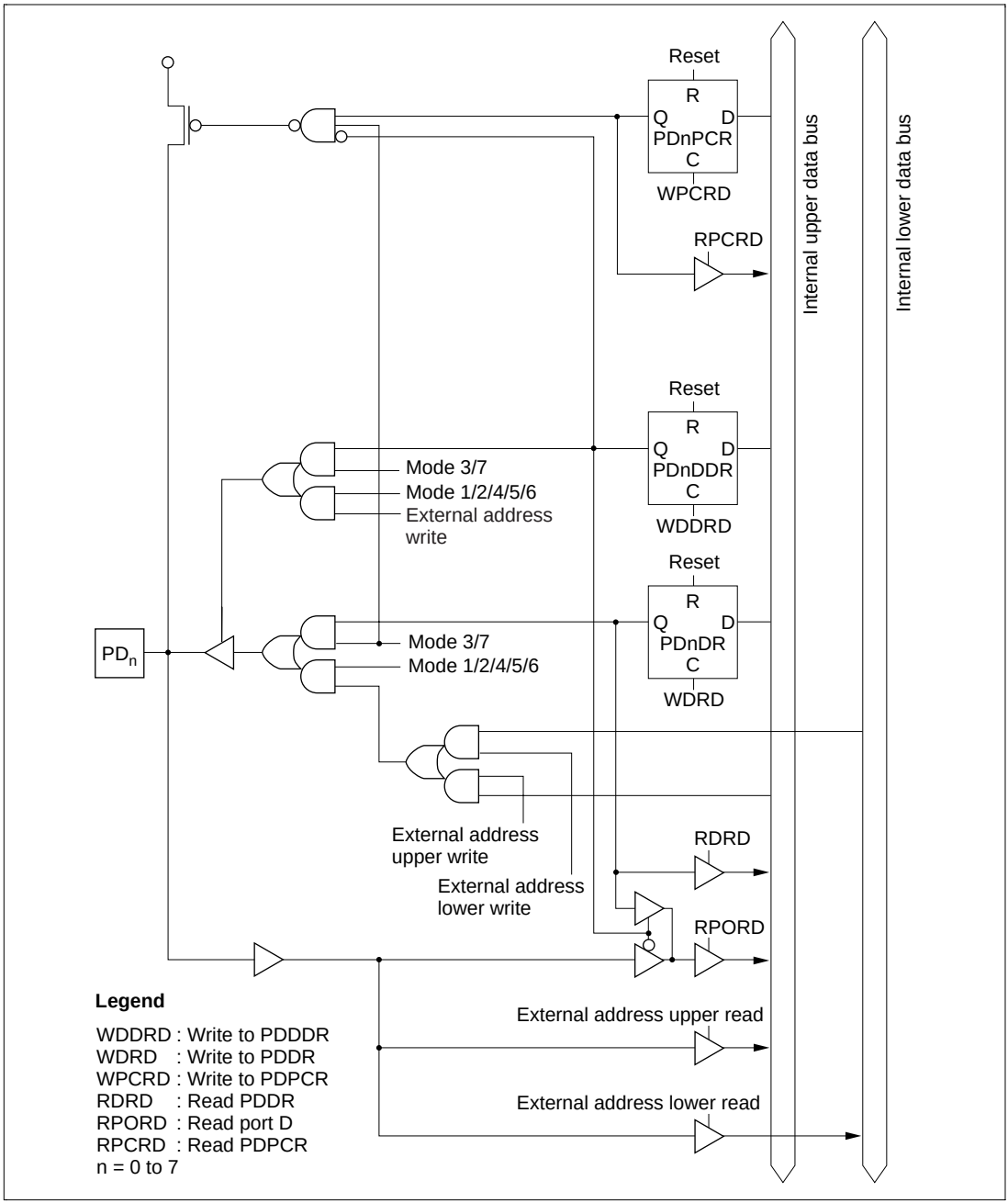



**Figure C-10 Port D Block Diagram (Pin $PD_n$)**

## C.11 Port E Block Diagram

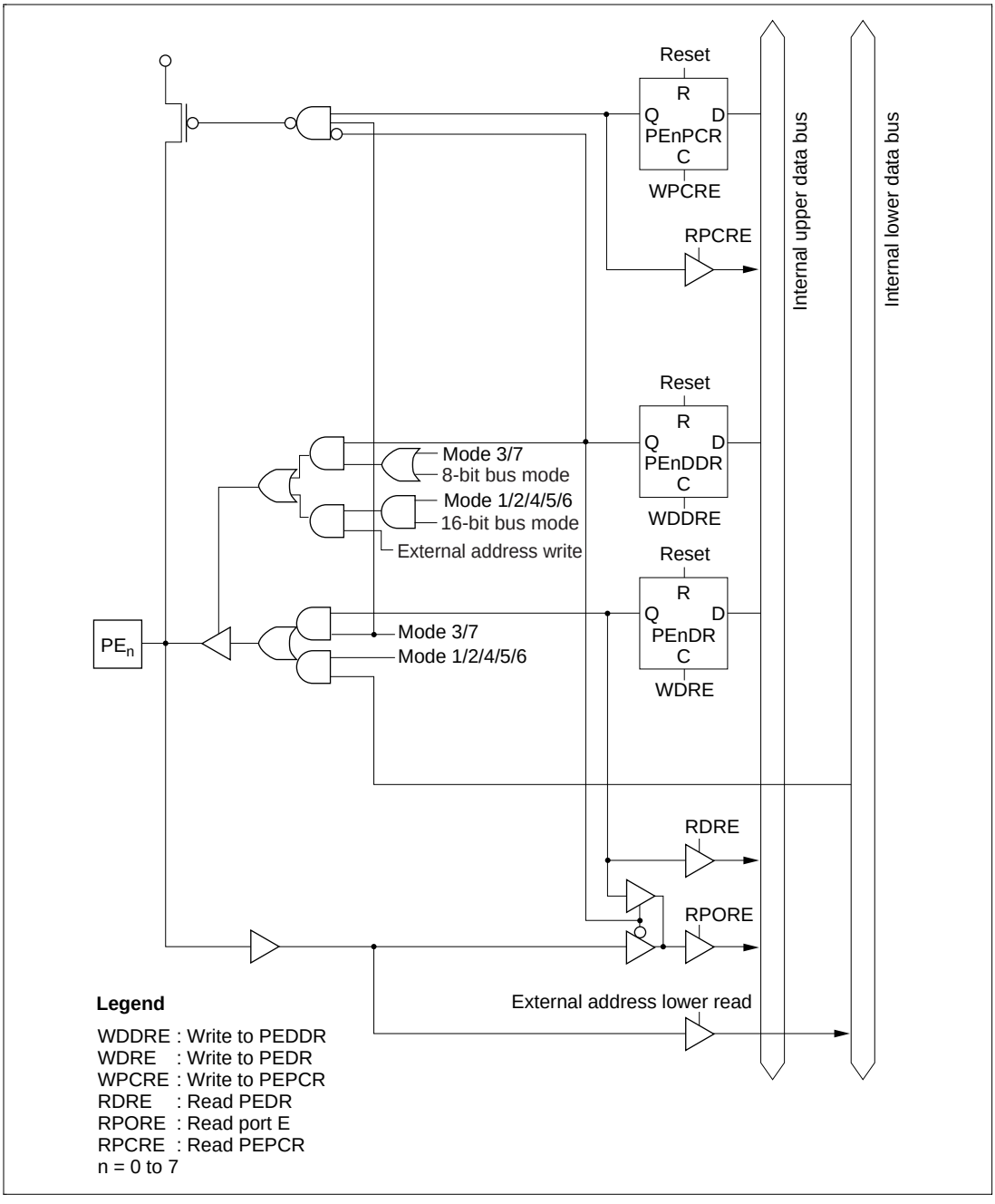



**Figure C-11 Port E Block Diagram (Pin $PE_n$)**

## C.12 Port F Block Diagram

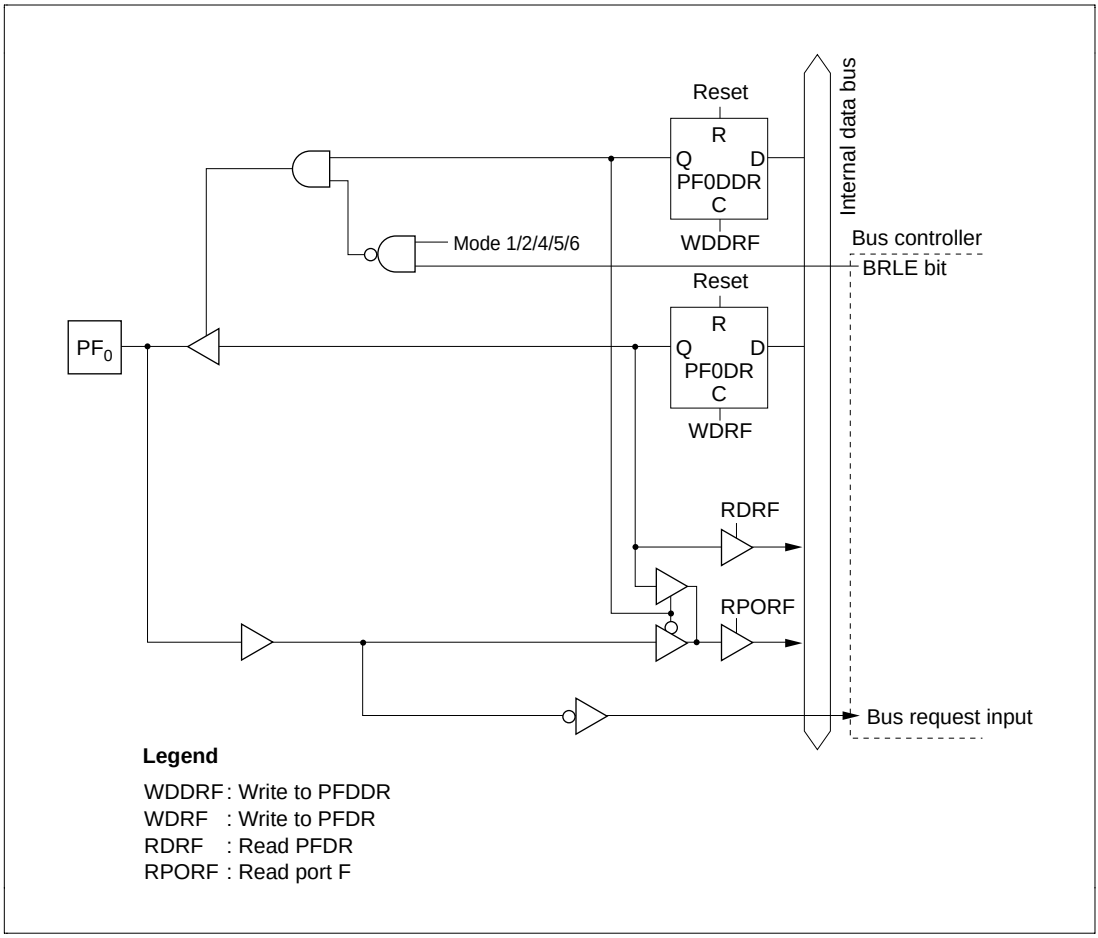



**Figure C-12 (a) Port F Block Diagram (Pin $PF_0$)**

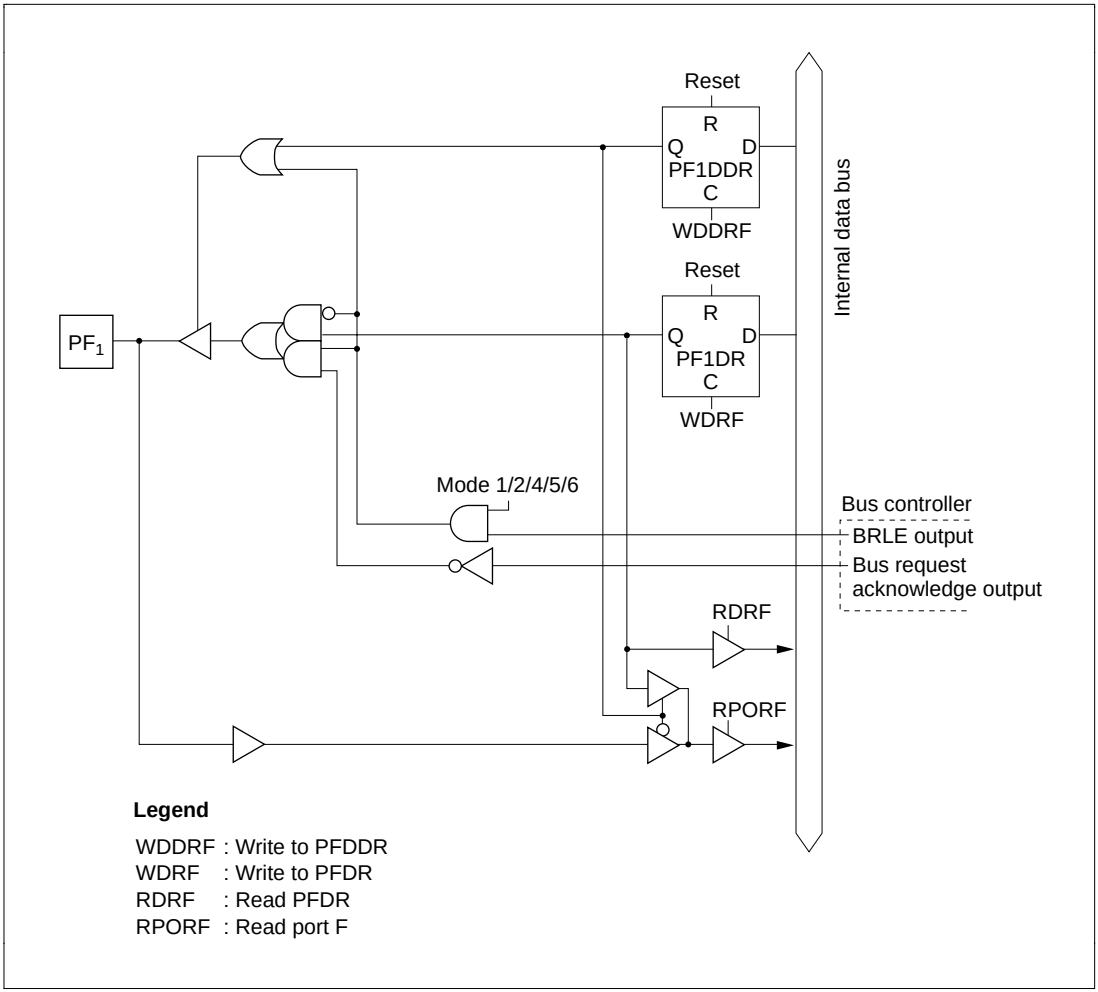



**Figure C-12 (b) Port F Block Diagram (Pin $PF_1$)**

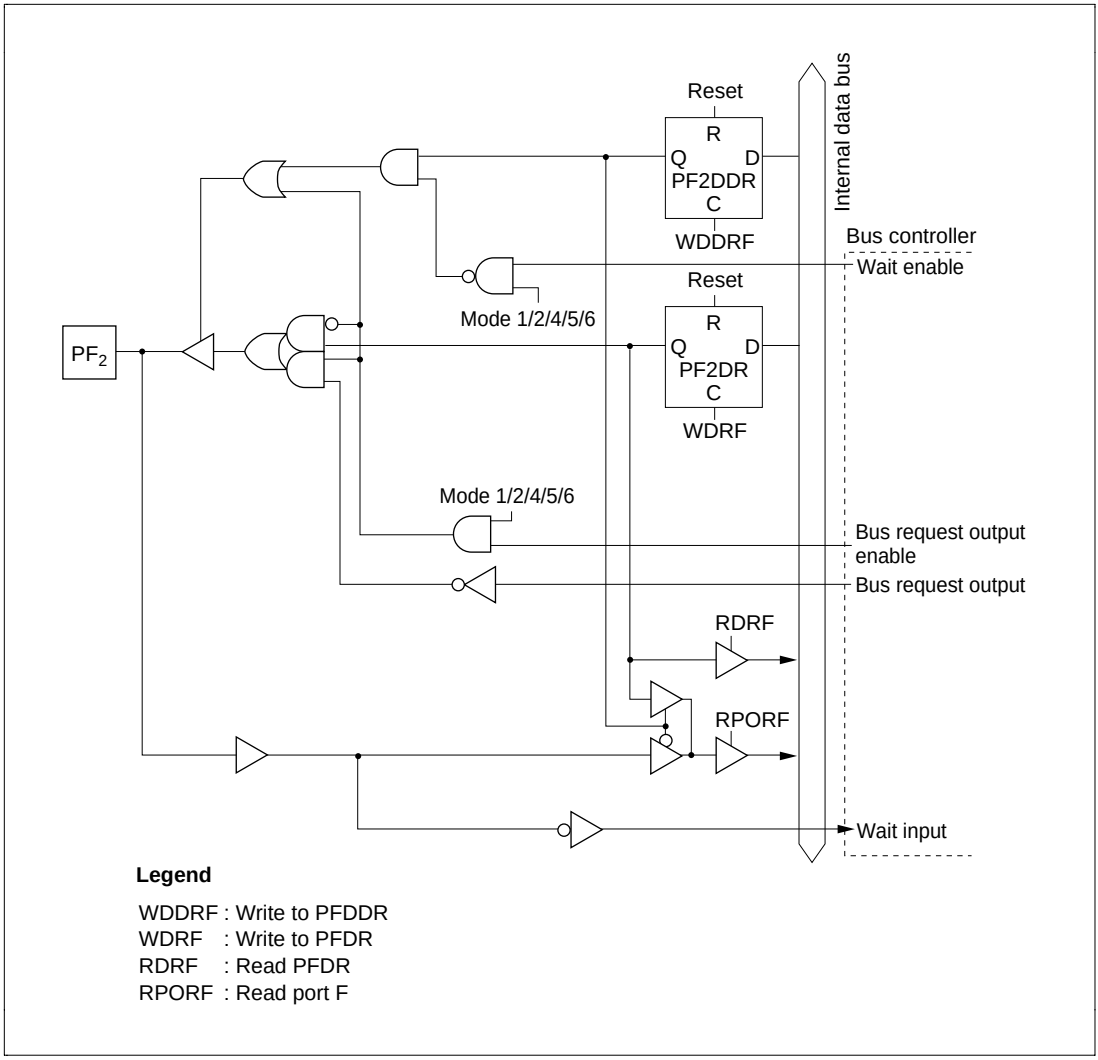



**Figure C-12 (c) Port F Block Diagram (Pin $PF_2$)**

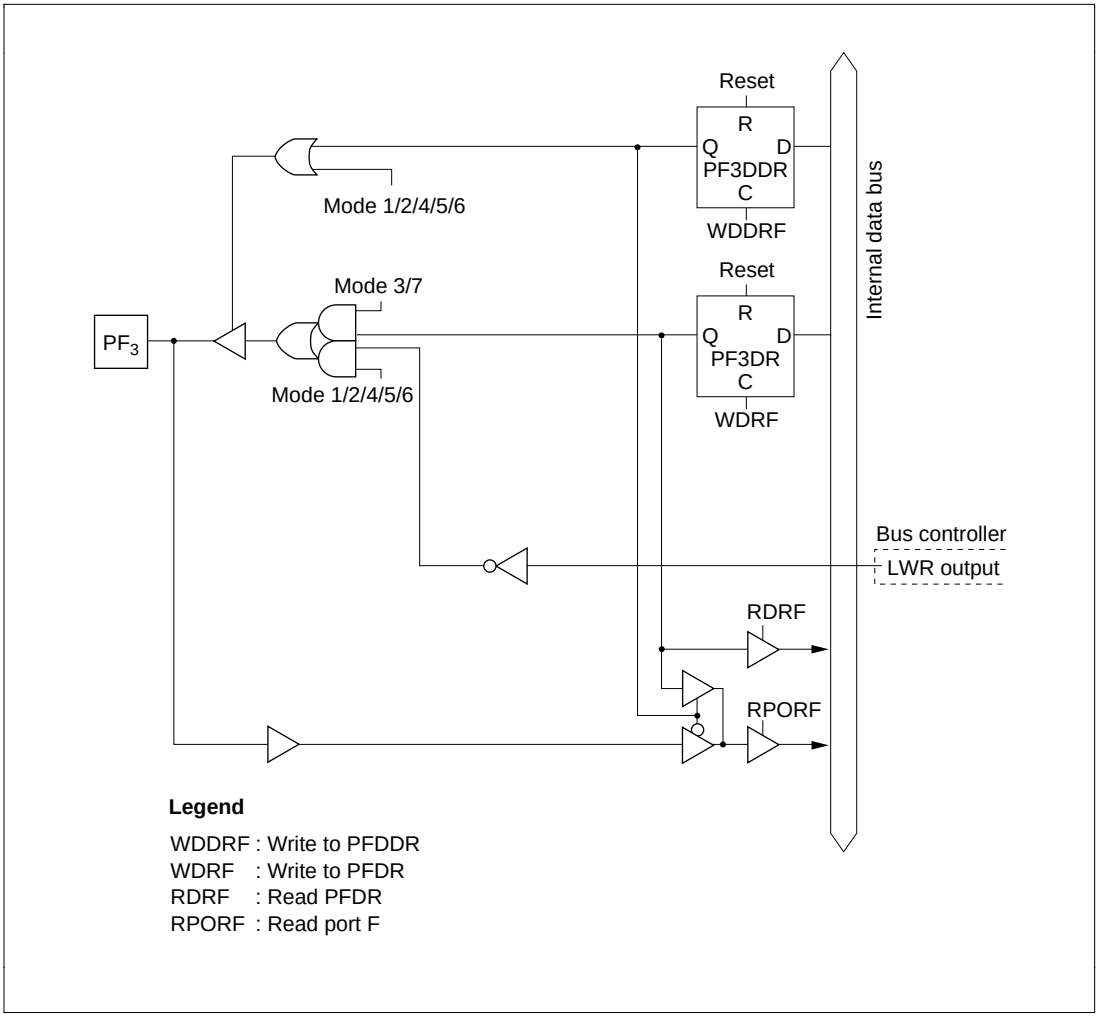



**Figure C-12 (d) Port F Block Diagram (Pin $PF_3$)**

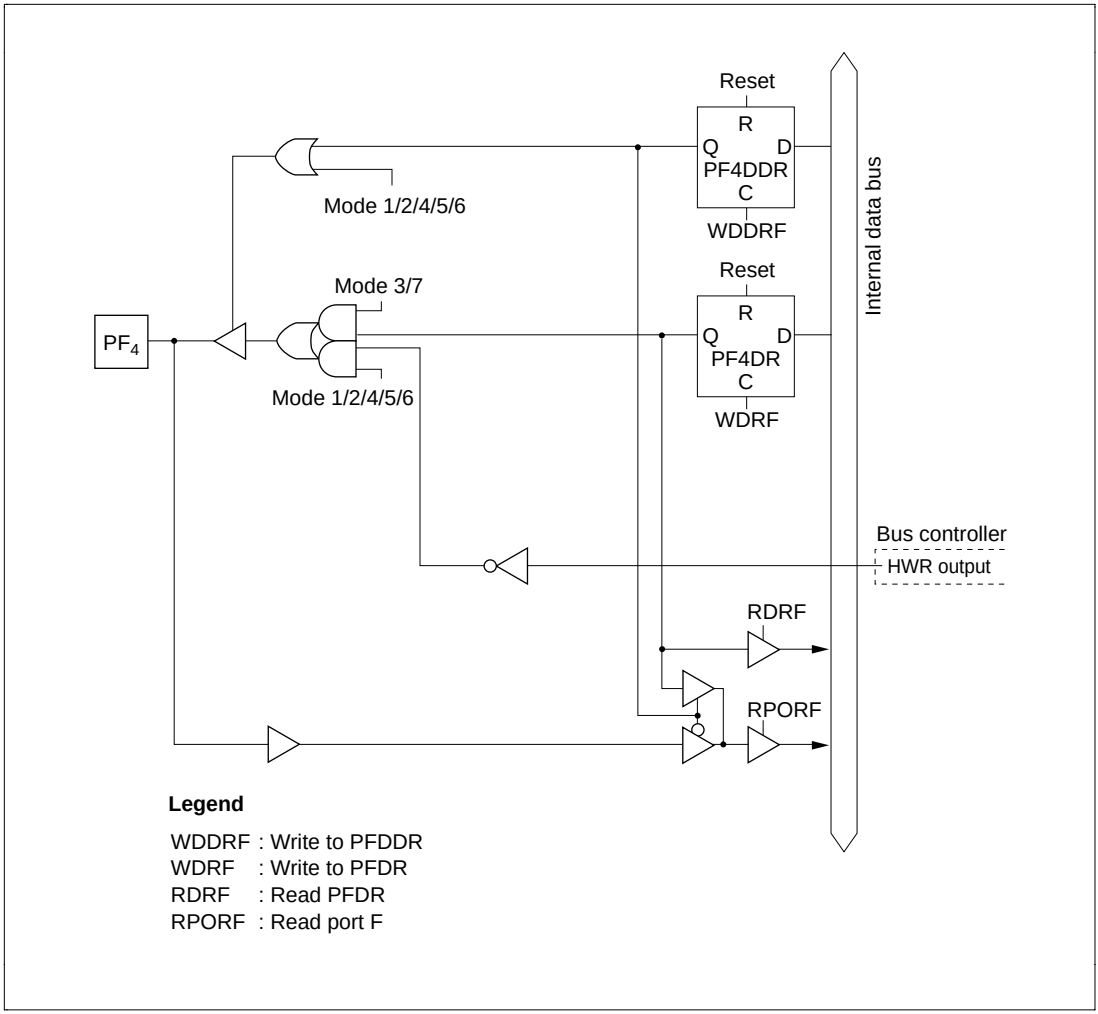



**Figure C-12 (e) Port F Block Diagram (Pin $PF_4$)**

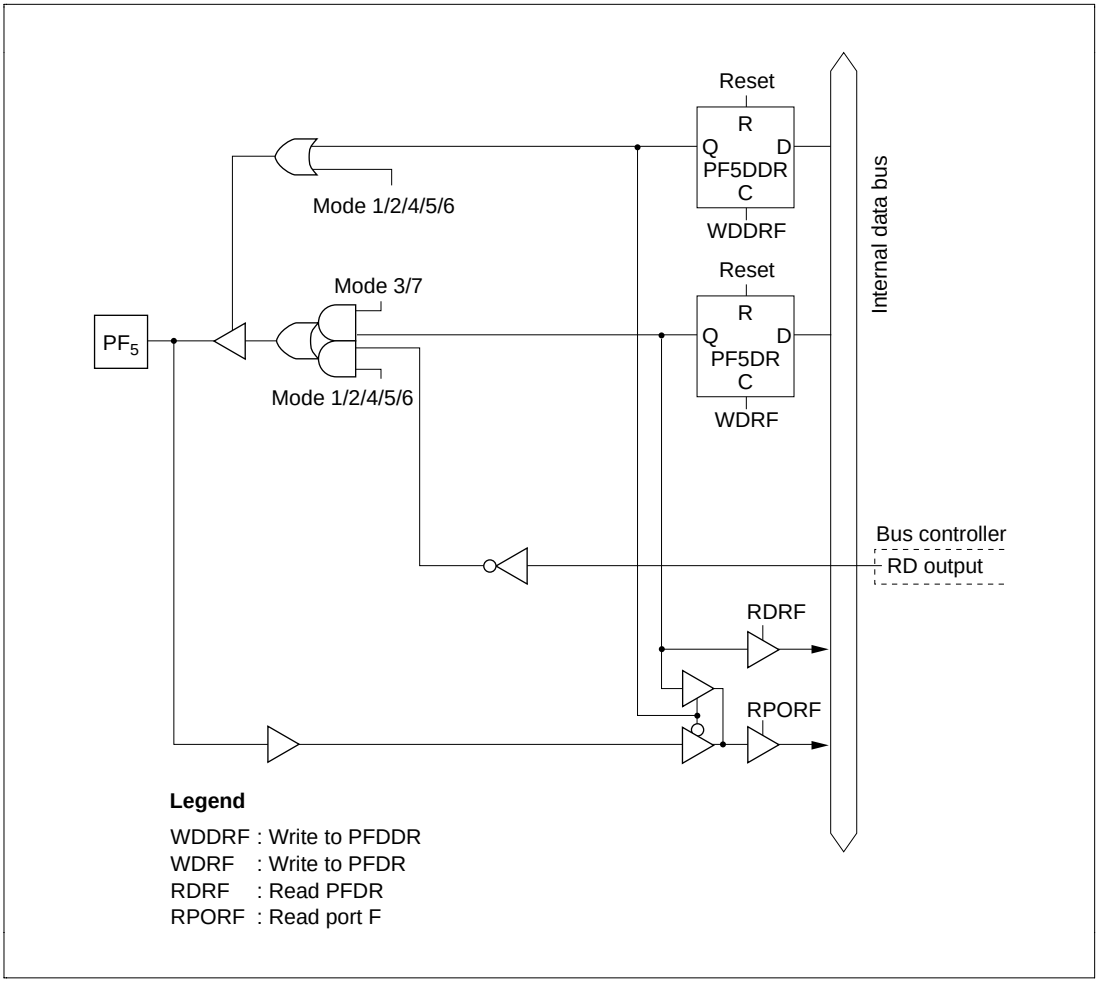



**Figure C-12 (f) Port F Block Diagram (Pin $PF_5$)**

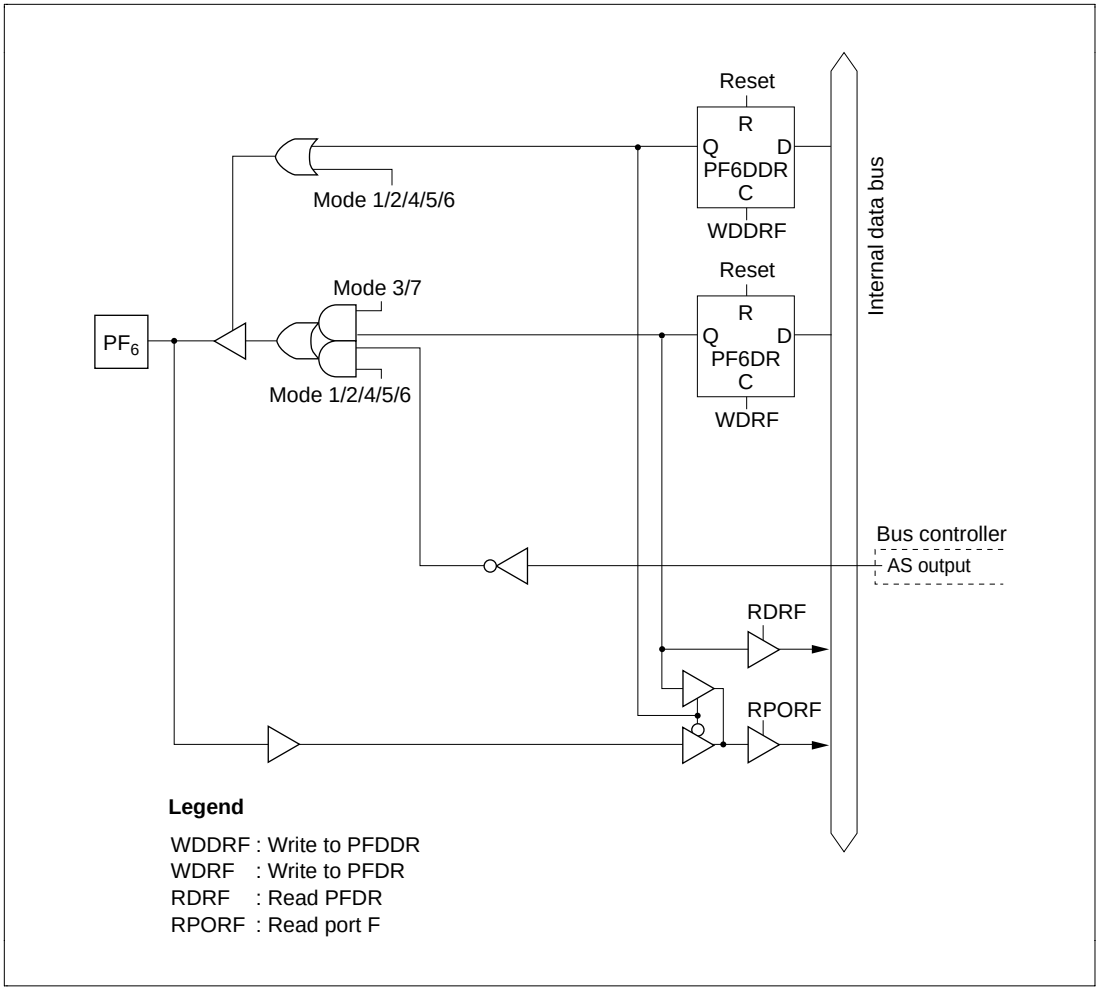



**Figure C-12 (g) Port F Block Diagram (Pin $PF_6$)**

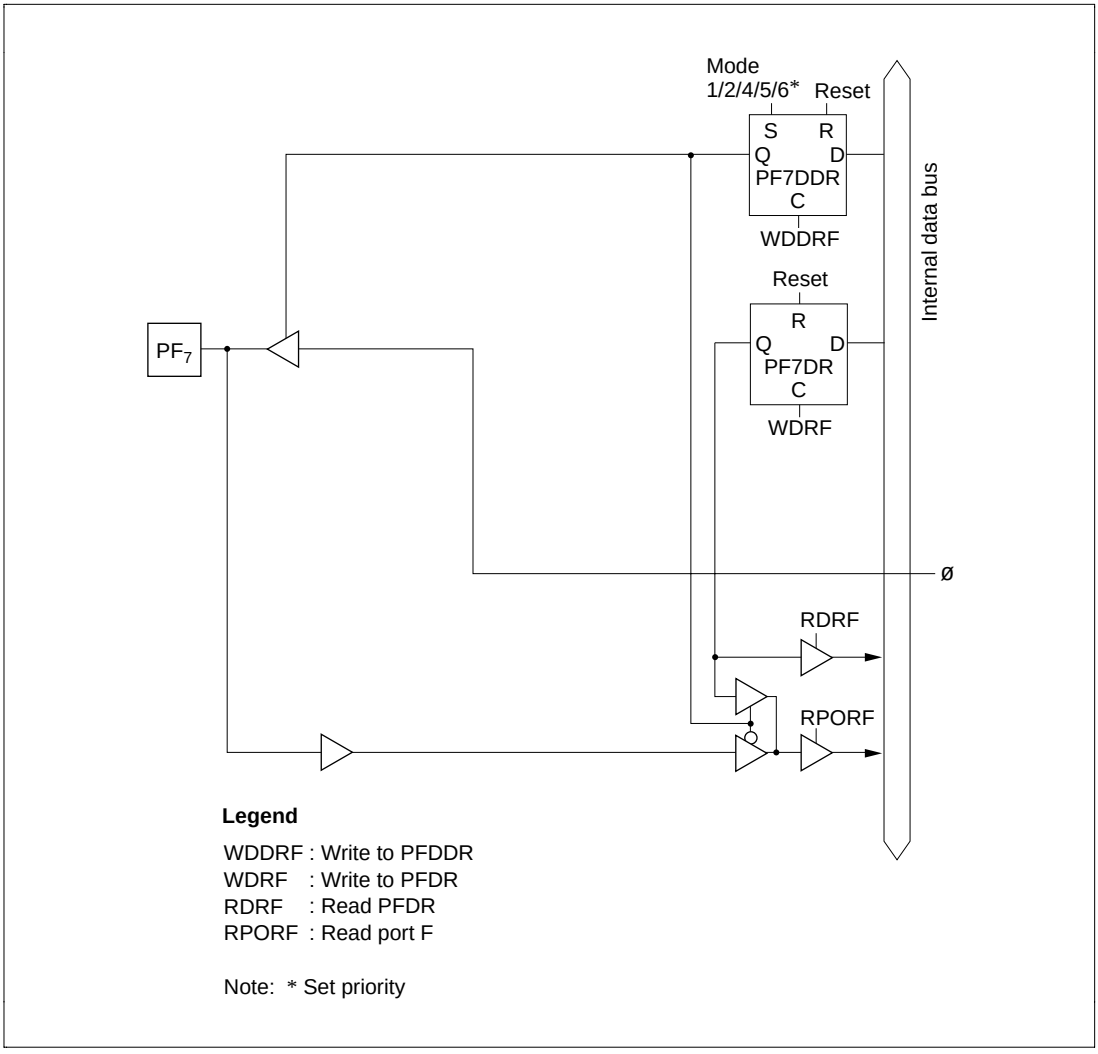



**Figure C-12 (h) Port F Block Diagram (Pin $PF_7$)**

## C.13 Port G Block Diagram

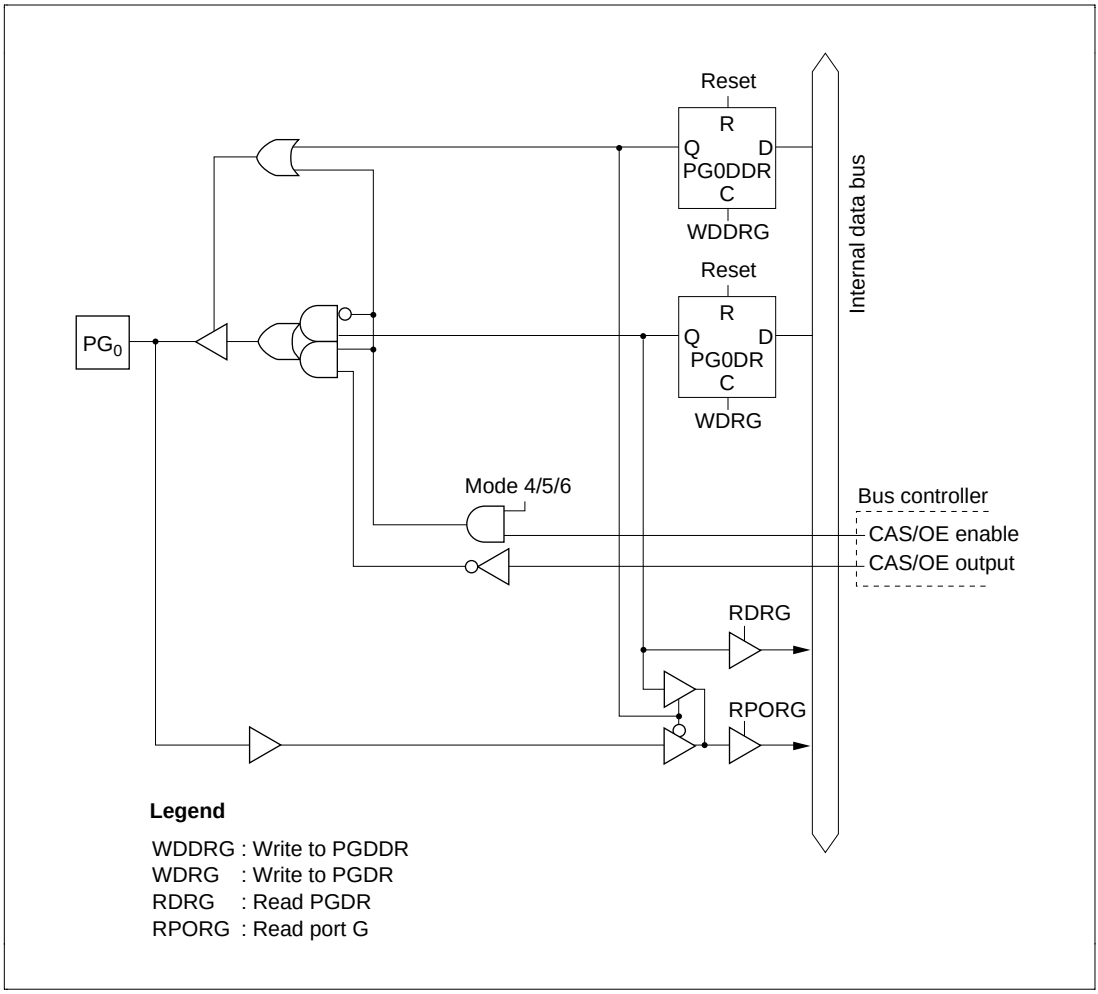



**Figure C-13 (a) Port G Block Diagram (Pin $PG_0$)**

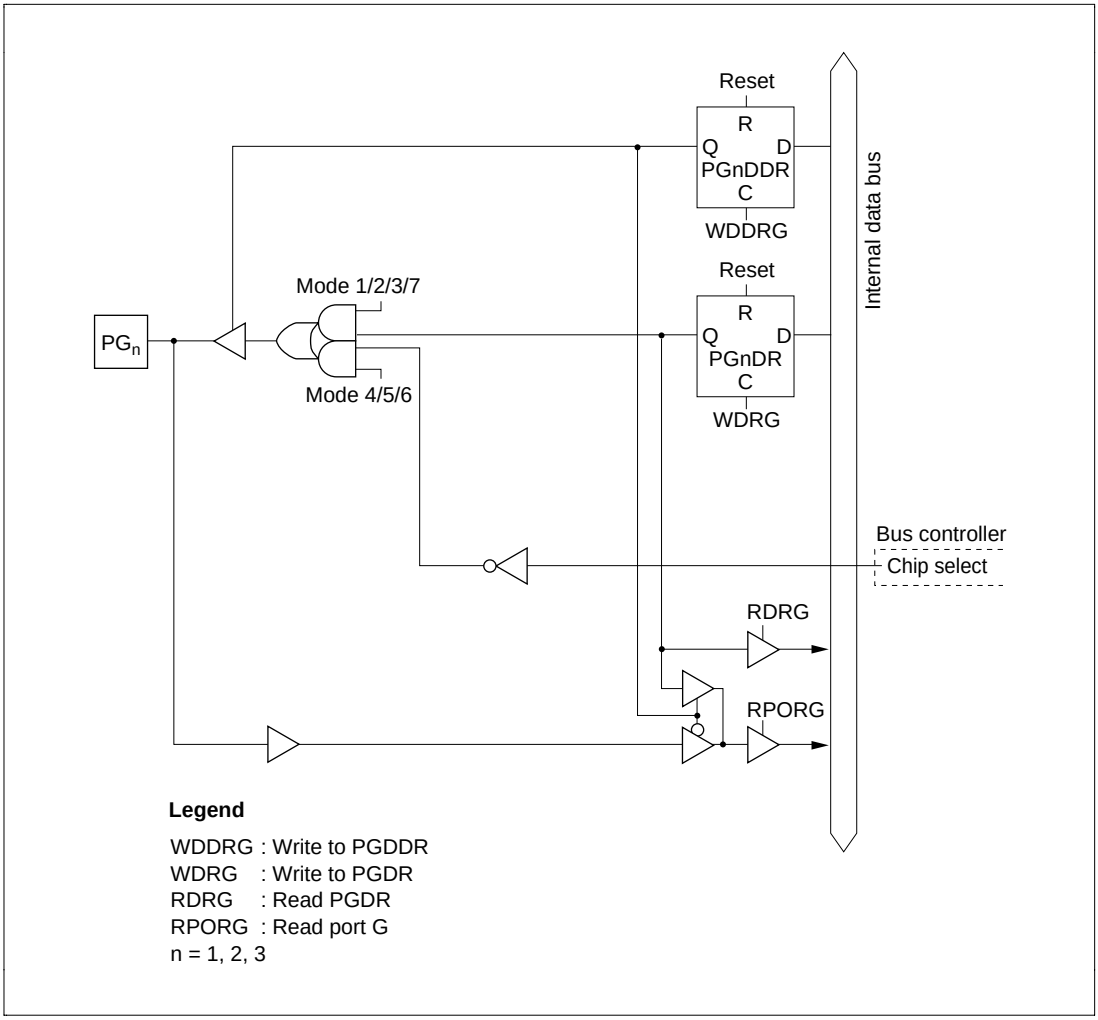



**Figure C-13 (b) Port G Block Diagram (Pins $PG_1$ to $PG_3$)**

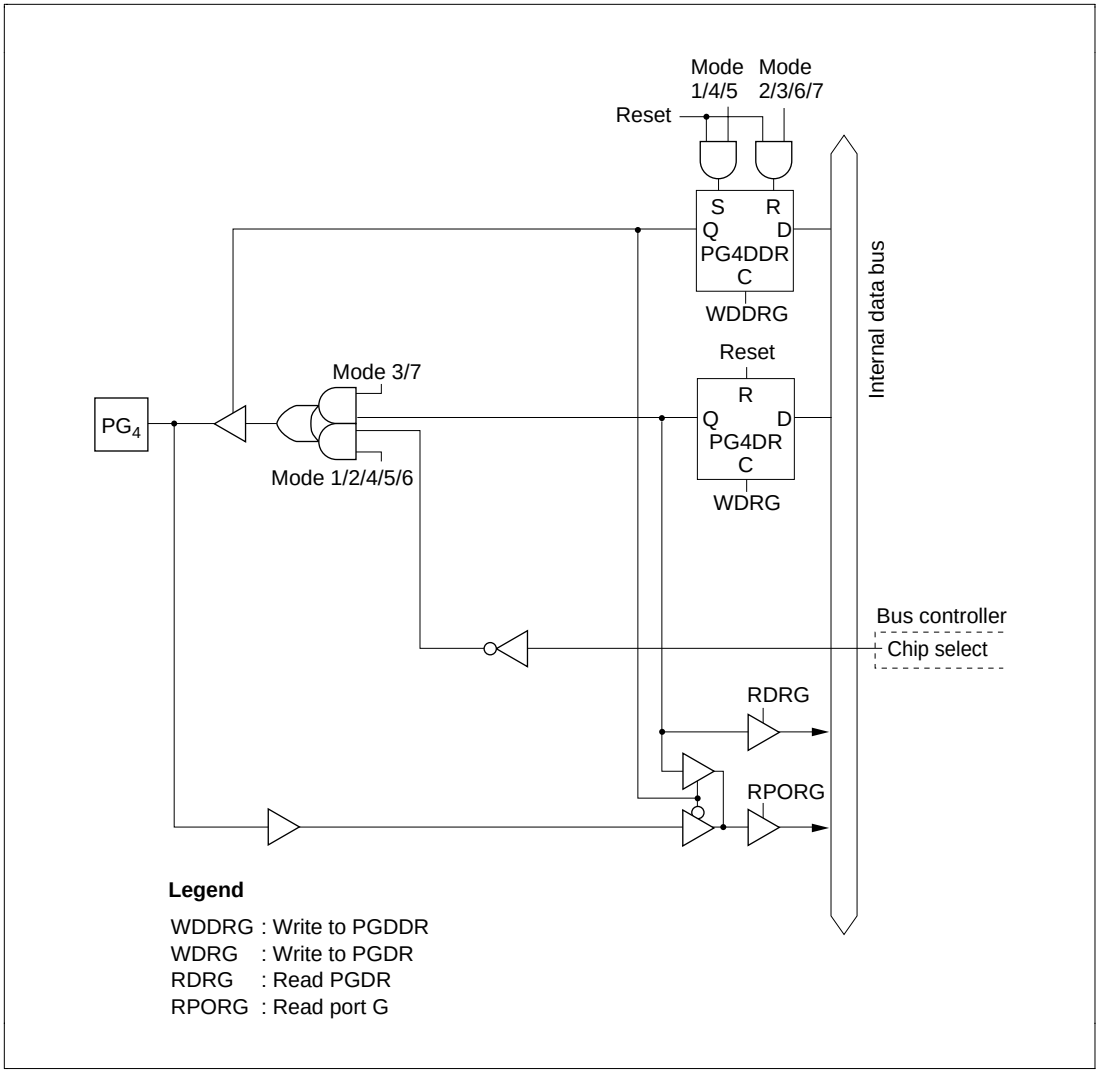



**Figure C-13 (c) Port G Block Diagram (Pin $PG_4$)**

# Appendix D Pin States

## D.1 Port States in Each Mode

**Table D-1 I/O Port States in Each Processing State**

| Port Name<br>Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State<br>Sleep Mode |
|---|---|---|---|---|---|---|---|
| Port 1 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| Port 2 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| Port 3 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| Port 4 | 1 to 7 | T | T | T | T | T | Input port |
| Port 5 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| $P6_5$ to $P6_2$ | 1 to 7 | T | kept | T | kept | kept | I/O port |
| $P6_7/\overline{CS_7}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| $P6_6/\overline{CS_6}$<br>$P6_1/\overline{CS_5}$<br>$P6_0/\overline{CS_4}$ | 4 to 6 | T | kept | T | [DDR · OPE = 0]<br>T<br>[DDR · OPE = 1]<br>H | T | [DDR = 0]<br>Input port<br>[DDR = 1]<br>$\overline{CS_7}$ to $\overline{CS_4}$ |
| Port A | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| PA 0 to PA3 | 4, 5 | L | kept | T | [OPE = 0]<br>T<br>[OPE = 1]<br>kept | T | Address output |
| | 6 | T | kept | T | [DDR · OPE = 0]<br>T<br>[DDR · OPE = 1]<br>kept | T | [DDR = 0]<br>Input port<br>[DDR = 1]<br>Address output |
| Port A | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| PA 4 | 4, 5 | L | kept | T | [OPE = 0]<br>T<br>[OPE = 1]<br>kept | T | Address output |
| | 6 | T | kept | T | [DDR · OPE = 0]<br>T<br>[DDR · OPE = 1]<br>kept | T | [DDR = 0]<br>Input port<br>[DDR = 1]<br>Address output |

**Table D-1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| Port A | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| PA 5 to PA7 | 4, 5 | T | kept | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] kept | T | [DDR = 0] Input port<br>[DDR = 1] Address output |
| | 6 | T | kept | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] kept | T | [DDR = 0] Input port<br>[DDR = 1] Address output |
| Port B | 1, 4, 5 | L | kept | T | [OPE = 0] T<br>[OPE = 1] kept | T | Address output |
| | 2, 6 | T | kept | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] kept | T | [DDR = 0] Input port<br>[DDR = 1] Address output |
| | 3, 7 | T | kept | T | kept | kept | I/O port |

**Table D-1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| Port C | 1, 4, 5 | L | kept | T | [OPE = 0] T [OPE = 1] kept | T | Address output |
| | 2, 6 | T | kept | T | [DDR · OPE = 0] T [DDR · OPE = 1] kept | T | [DDR = 0] Input port [DDR = 1] Address output |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| Port D | 1, 2, 4 to 6 | T | T* | T | T | T | Data bus |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| Port E | 1, 2, 4 to 6 (8 bit bus) | T | kept | T | kept | kept | I/O port |
| | 1, 2, 4 to 6 (16 bit bus) | T | T* | T | T | T | Data bus |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_7$/ø | 1, 2, 4 to 6 | Clock output | [DDR = 0] T [DDR = 1] Clock output | T | [DDR = 0] Input port [DDR = 1] H | [DDR = 0] Input port [DDR = 1] Clock output | [DDR = 0] Input port [DDR = 1] Clock output |
| | 3, 7 | T | kept | T | [DDR = 0] Input port [DDR = 1] H | [DDR = 0] Input port [DDR = 1] Clock output | [DDR = 0] Input port [DDR = 1] Clock output |
| $PF_6$/$\overline{AS}$ $PF_5$/$\overline{RD}$ $PF_4$/$\overline{HWR}$ $PF_3$/$\overline{LWR}$ | 1, 2, 4 to 6 | H | H* | T | [OPE = 0] T [OPE = 1] H | T | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |

**Table D-1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| $PF_2$/$\overline{LCAS}$/ $\overline{WAIT}$/ $\overline{BREQO}$ | 1, 2, 4 to 6 | T | [BREQOE + WAITE + LCASE = 0] kept<br>[BREQOE = 1] $\overline{BREQO}$<br>[WAITE = 1] T<br>[LCASE = 1] H* | T | [BREQOE + WAITE + LCASE = 0] kept<br>[BREQOE = 1] kept<br>[WAITE = 1] T<br>[LCASE = 1, OPE = 0] T<br>[LCASE = 1, OPE = 1] $\overline{LCAS}$ | [BREQOE + WAITE + LCASE = 0] kept<br>[BREQOE = 1] $\overline{BREQO}$<br>[WAITE = 1] T<br>[LCASE = 1] T | [BREQOE + WAITE + LCASE= 0] I/O port<br>[BREQOE = 1] $\overline{BREQO}$<br>[WAITE = 1] $\overline{WAIT}$<br>[LCASE = 1] $\overline{LCAS}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_1$/$\overline{BACK}$ | 1, 2, 4 to 6 | T | [BRLE = 0] kept<br>[BRLE = 1] $\overline{BACK}$ | T | [BRLE = 0] kept<br>[BRLE = 1] $\overline{BACK}$ | L | [BRLE = 0] I/O port<br>[BRLE = 1] $\overline{BACK}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_0$/$\overline{BREQ}$ | 1, 2, 4 to 6 | T | [BRLE = 0] kept<br>[BRLE = 1] $\overline{BREQ}$ | T | [BRLE = 0] kept<br>[BRLE = 1] T | T | [BRLE = 0] I/O port<br>[BRLE = 1] $\overline{BREQ}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PG_4$/$\overline{CS_0}$ | 1, 4, 5 | H | [DDR = 0] T<br>[DDR = 1] H* | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] H | T | [DDR = 0] Input port<br>[DDR = 1] $\overline{CS_0}$ |
| | 2, 6 | T | | | | | |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PG_3$/$\overline{CS_1}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| $PG_2$/$\overline{CS_2}$<br>$PG_1$/$\overline{CS_3}$ | 4 to 6 | T | [DDR = 0] T<br>[DDR = 1] H* | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] H | T | [DDR = 0] Input port<br>[DDR = 1] $\overline{CS_1}$ to $\overline{CS_3}$ |

**Table D-1 I/O Port States in Each Processing State (cont)**

| Port Name<br>Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State<br>Sleep Mode |
|---|---|---|---|---|---|---|---|
| $\overline{PG_0}/\overline{CAS}$<br>$\overline{OE}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| | 4 to 6 | T | [DRAME = 0, PSRAME = 0]<br>kept<br>[DRAME = 1]<br>H*<br>[PSRAME = 1]<br>H* | T | [DRAME = 0, PSRAME = 0]<br>kept<br>[OPE = 0]<br>T<br>[DRAME · OPE= 1]<br>$\overline{CAS}$<br>[PSRAME · OPE = 1]<br>$\overline{OE}$ | T | [DRAME = 0, PSRAME = 0]<br>Input port<br>[DRAME = 1, PSRAME = 0]<br>$\overline{CAS}$<br>[DRAME = 0, PSRAME = 1]<br>$\overline{OE}$ |

**Legend:**

H : High level
L : Low level
T : High impedance
kept : Input port becomes high-impedance, output port retains state
DDR : Data direction register
OPE : Output port enable
WAITE : Wait input enable
BRLE : Bus release enable
BREQOE : BREQO pin enable
DRAME : DRAM space setting
LCASE : DRAM space setting, CW2 = LCASS = 0
PSRAME : PSRAM space setting

Note: * Indicates the state after completion of the executing bus cycle.

# Appendix E Timing of Transition to and Recovery from Hardware Standby Mode

## Timing of Transition to Hardware Standby Mode

(1) To retain RAM contents with the RAME bit set to 1 in SYSCR, drive the $\overline{\text{RES}}$ signal low at least 10 states before the $\overline{\text{STBY}}$ signal goes low, as shown below. $\overline{\text{RES}}$ must remain low until $\overline{\text{STBY}}$ signal goes low (delay from $\overline{\text{STBY}}$ low to $\overline{\text{RES}}$ high: 0 ns or more).

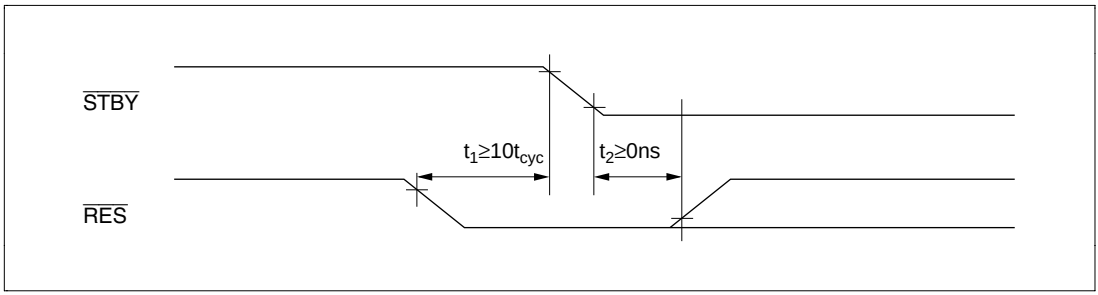



**Figure E-1 Timing of Transition to Hardware Standby Mode**

(2) To retain RAM contents with the RAME bit cleared to 0 in SYSCR, or when RAM contents do not need to be retained, $\overline{\text{RES}}$ does not have to be driven low as in (1).

## Timing of Recovery from Hardware Standby Mode

Drive the $\overline{\text{RES}}$ signal low and the NMI signal high approximately 100 ns or more before $\overline{\text{STBY}}$ goes high to execute a power-on reset.

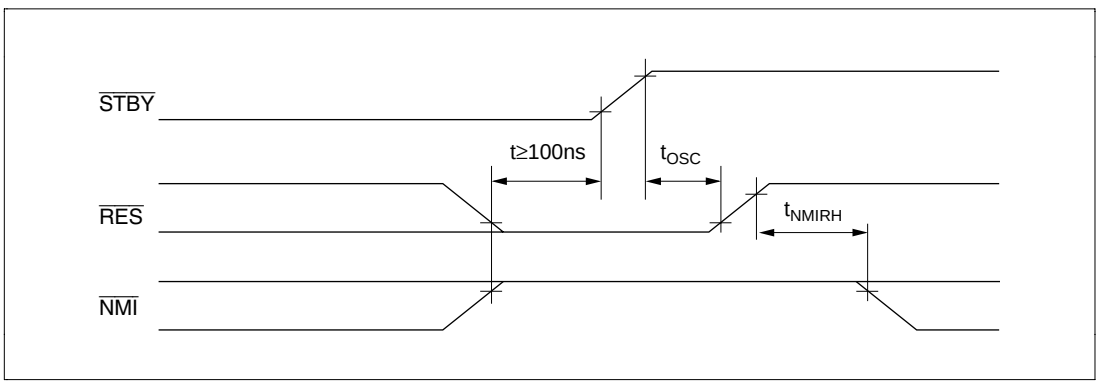



**Figure E-2 Timing of Recovery from Hardware Standby Mode**

# Appendix F Product Code Lineup

**Table F.1 H8S/2655 Series Product Code Lineup**

| Product Type | | | Product Code | Mark Code | Package (Hitachi Package Code) |
|---|---|---|---|---|---|
| H8S/2655 | Mask ROM | 5 V version ($V_{CC}$=5.0V±10%) | HD6432655 | HD6432655(***)TE | 120-pin TFP (TFP-120) |
| | | | | HD6432655(***)F | 128-pin FP (FP-128) |
| | | Low-voltage version ($V_{CC}$=2.7 to 5.5V) | | HD6432655(***)TE | 120-pin TFP (TFP-120) |
| | | | | HD6432655(***)F | 128-pin FP (FP-128) |
| | ZTAT™ | 5 V version ($V_{CC}$=5.0V±10%) | HD6472655 | HD6472655TE | 120-pin TFP (TFP-120) |
| | | | | HD6472655F | 128-pin FP (FP-128) |
| | | Low-voltage version ($V_{CC}$=2.7 to 5.5V) | | HD6472655VTE | 120-pin TFP (TFP-120) |
| | | | | HD6472655VF | 128-pin FP (FP-128) |
| H8S/2653 | Mask ROM | 5 V version ($V_{CC}$=5.0V±10%) | HD6432653 | HD6432653(***)TE | 120-pin TFP (TFP-120) |
| | | | | HD6432653(***)F | 128-pin FP (FP-128) |
| | | Low-voltage version ($V_{CC}$=2.7 to 5.5V) | | HD6432653(***)TE | 120-pin TFP (TFP-120) |
| | | | | HD6432653(***)F | 128-pin FP (FP-128) |

Note: (***) indicates the ROM code.

# Appendix G Package Dimensions

Figures G-1 and G-2 show the TFP-120 and FP-128 package dimensions of the H8/2655 Series.

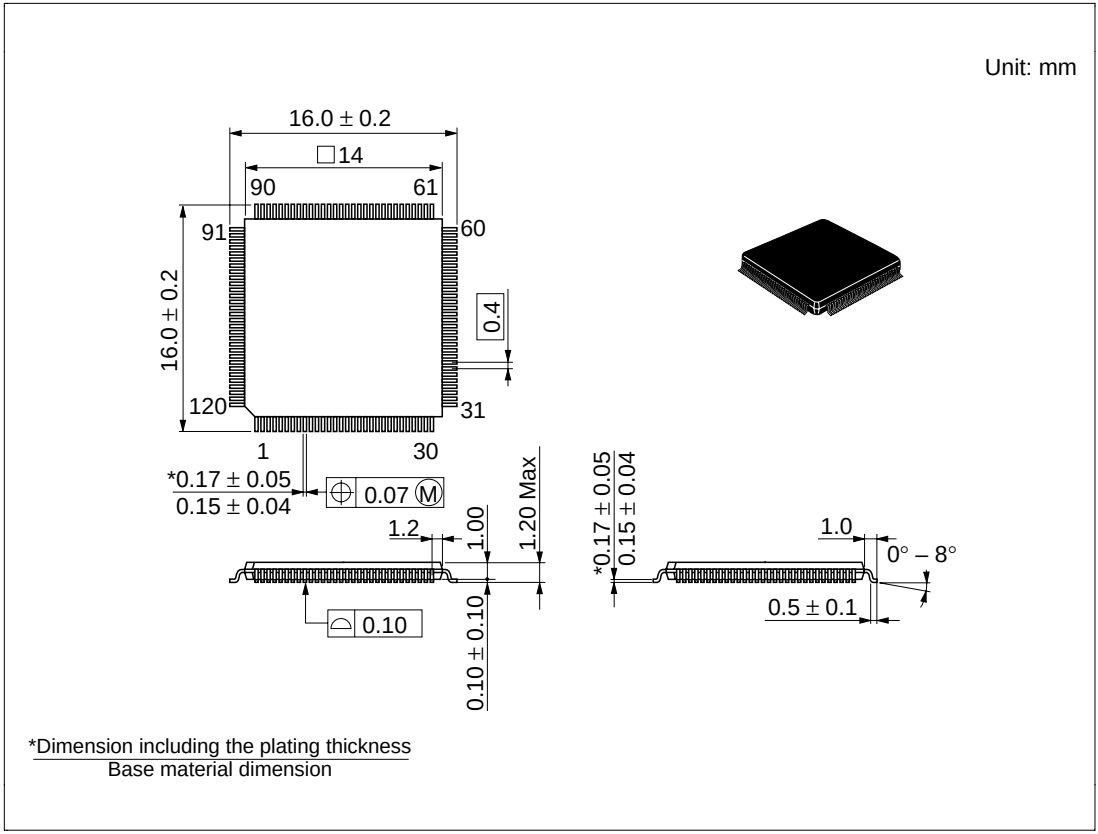



**Figure G-1 TFP-120 Package Dimensions**

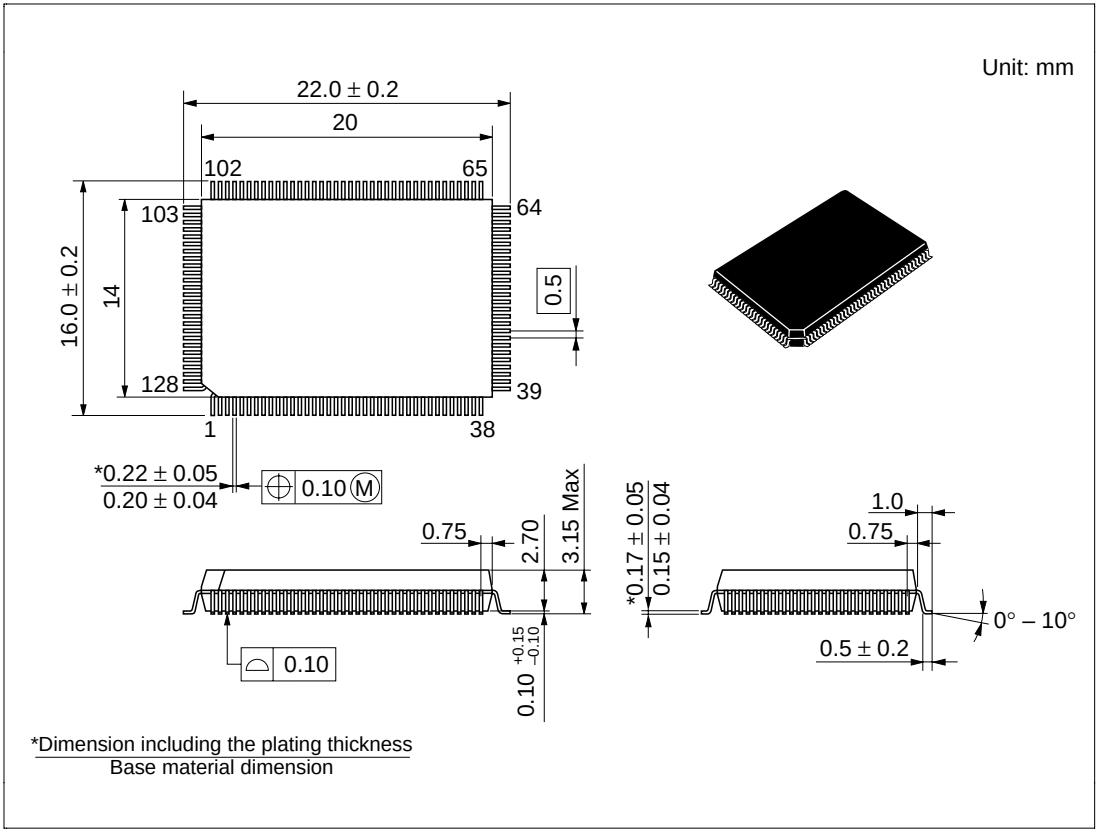



**Figure G-2 FP-128 Package Dimensions**

**H8S/2655 Series Hardware Manual**

Publication Date: 1st Edition, November 1995
4th Edition, September 1998
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.
Copyright © Hitachi, Ltd., 1995. All rights reserved. Printed in Japan.